Raj R. Singh

Dissertation Proposal:

Distributed and Collaborative Planning Analytics in a Networked Society

Dissertation Committee:

Head: ______________________________

Joseph Ferreira, Jr.

Professor of Urban Planning and Operations Research, MIT

Member: ______________________________

Lewis Hopkins

Professor of Urban and Regional Planning and
Landscape Architecture, University of Illinois

Member: ______________________________

Lorlene Hoyt

Assistant Professor of Technology and Planning, MIT

April 21, 2004

TABLE OF CONTENTS

INTRODUCTION

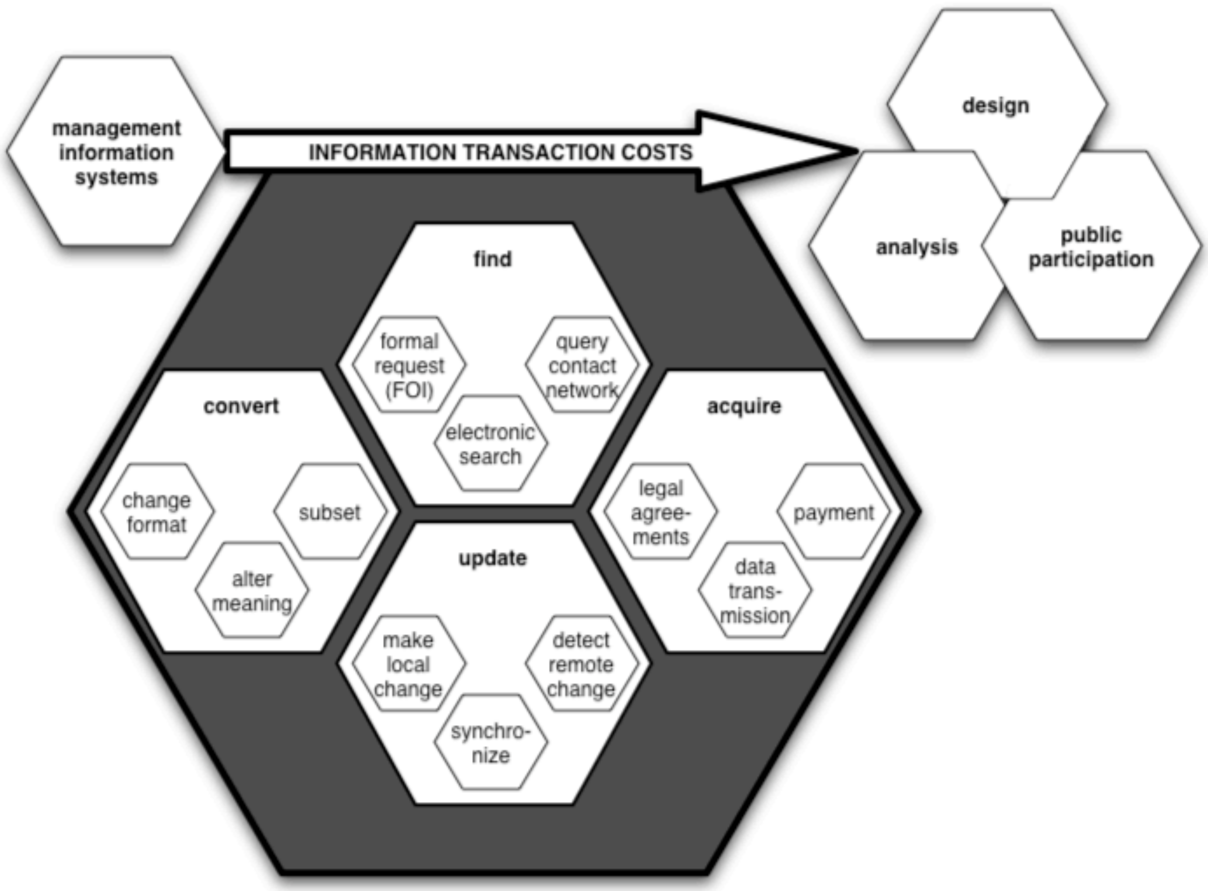
The field of Planning Support Systems (PSS) has "matured into a conception of integrated systems of information and software which bring the three components of traditional decision support systems—information, models, and visualization—into the public realm (Batty, 1995; Klosterman, 1997)." This definition is taken from Klosterman's guest editorial in the 1999 *Environment and Planning B* theme issue on PSS. While I agree with him on the definition, I do not agree that our 'conception of integrated systems of information and software' is mature. It is, rather, immature in that PSS has few answers to what I believe is one of the most important information integration challenges facing us today—*reducing the high cost of informed decision making*.

Urban environments have become such incredibly complex organisms that no single person or agency has the data or knowledge upon which to make responsible decisions. Instead, we rely on a web of organizations to build and maintain databases and analytic tools that can be brought to bear on planning problems. The problem is that it is extremely 'expensive' to transfer information from one entity to another, whether that entity is another person, organization, or computer. The expense will usually take the form of labor (time), although other common costs are complexity (expertise), legal barriers, and technology. The result is that planning decisions are based upon information that is 'cheap' to acquire and use. Or in rare cases, we go to great expense to make a well-informed decision, then have no resources to go back and re-analyze the problem at a later date.

Sovling this problem becomes even more critical as we enter a time when billions of dollars will be spent on digital, or e-government. The World Wide Web has attracted the attention of all levels of government in the U.S., from the federal E-Government Act of 2002 to the National Civic League's 8th revision of the Model City Charter. The question is how should government leverage electronic networks to improve operations? If we define the Web narrowly as a public face on information sources, we will end up mimicking the systems of today, wasting an opportunity to rethink basic computing and information sharing paradigms in light of network computing. Corporations were asking themselves these same questions in the 1990s, and responding to the challenges of a digitally connected business environment led to major paradigm shifts at all levels of the enterprise. I believe the same thing will take place in the public sector, offering researchers an excellent opportunity to affect dramatic change in the way government administers, archives, and shares information. But urban planning is not the corporate world, and we must respond to these challenges in our own way if we want to avoid another generation of using business technology to solve planning problems.

RESEARCH PLAN

Rethinking the conception of integrated information systems is a large, far-reaching problem to tackle, so it is important to carefully choose as small a subset of the problem as possible while retaining enough generality to have broad applicability in the field. Britton Harris' (Harris, 1999) described planning as being comprised of three major professional activities—analysis, design, and public participation. One sensible way of narrowing my problem would be to study the information flow between two of these activities. In fact, Harris himself argues that we have been weak at integrating design and public participation. However, I feel that too many of the opportunities for improvement have already been lost by the time information makes its way into the recognized planning activities. To have the most impact, it is best to include a fourth activity, the administration of public records, which I will refer to using the generic term Management Information Systems (MIS). Planners often neglect this group because they are really part of another profession, but activities such as property assessment, permitting, surveying, cable laying, etc. comprise our main information resources, and therefore must be an integral part of our thinking. So, my simplification of the problem is to focus on the information transactions between MIS and PSS.



My strategy for undertaking this primarily theoretical portion of the research is to start with the proposition that the efficacy of planning support systems can be greatly enhanced by increasing the quality and quantity of information available to them. I recognize that this is not a widely held point of view. In fact, much research has a built-in implicit assumption that the quality and quantity of data available to planners is fixed, and the work of PSS is to innovate around such things as user interfaces and statistical measures. So my starting point may already be disruptive, but like most theories, its value is in its ability to solve difficult problems, and is therefore hard to prove in any way but through empirical work.

In this dissertation I will begin this work with a three-point research plan. First, I propose an approach to the problem based on a theory of transaction costs and barriers to information flow. A great deal of progress can be made by identifying situations where the cost of getting information into planning support systems is high or even prohibitive. If it is possible to greatly reduce these costs (in time, money or type of personnel), PSS will see an increase in information content.

The second stage of my research is to work through a thought experiment. In the context of the MassGIS statewide buildout study, I will develop evaluation metrics that describe and quantify the types of transaction costs found in a typical planning analysis scenario. Based on the results, I will design a system comprised of policy and technology recommendations that greatly reduces those costs. The MassGIS study was conducted under the Community Preservation Act. Enacted in December 2000, this effort seeks to, "promote smarter land use to preserve and enhance the quality of life in communities across the Commonwealth." (*Buildout Book*, 2001). The buildout analysis is meant as a tool to encourage debate around development futures as much as to be a tool to accurately predict population or development pressures under current zoning. It serves as an ideal example of a typical physical planning model, because it is intended to stimulate public participation in decision making around a future that can best be predicted using a wealth of current information about the region in conjunction with the expertise of planning professionals. The buildout work, which was designed from a regional perspective, affords the opportunity to look at the analysis in relationship to the local administrative MIS-type professionals who manage property and environmental records, as well as the local planners that must use the study's results in concert with public opinion to develop new local land use policy.

This theory of reducing transaction costs to improve business processes is not novel. Commercial industries have pursued this line of research for decades. Recently, businesses like Amazon owe their existence to supply chain integration processes that minimize transaction costs between trading partners. The work of applying this research to the planning field is not trivial, because the nature of planning is quite different from corporations, where top-down decision making is the norm, and technology initiatives have a simple evaluation criterion—higher profits. In planning we must take a more democratic and holistic approach. We can examine more deeply not only the MIS-to-PSS-to-decision maker chain that resembles a business supply chain so closely. There is also an important horizontal audience—those analysts from different professions and analytic traditions that work on a problem and who often contribute perspectives that require significant re-formulation of planning problems—more significant changes than tweaking a few parameters within a fixed modeling structure—and who each tend to focus on related, but different, models and metrics. These include architects, hydrologists, economists, psychogeographers, and community activists. As Hopkins argues in "Structure of a planning support system for urban development" (1999), we lack an underlying structure with which to integrate all the efforts and analyses that constitute planning, from writing plans to holding meetings, to drawing designs and sketch plans to running simulations. There is no unifying framework to organize these efforts.

Using the experience of the buildout analysis in conjunction with the methodology I develop to evaluate transaction costs, the third and final stage of the work will be to design a strategy for reducing these costs without losing the highly important uniqueness of planning support systems—their "answers" are less important than their ability to shape and stimulate discussion around alternate futures, and inform equitable, timely decision making processes. The technological underpinnings of this strategy are described below.

Leveraging important technology trends

This research is inextricably linked to a number of fundamental changes I see happening in how government collects, stores and distributes data, and how Internetaware software is built. While planning cannot adopt corporate technology wholesale, unlike the military industry, we do not have the financial resources to develop our own basic technologies. This puts us in the precarious position of strategically choosing which technologies to adopt from other fields, and which ones we should develop ourselves. I list here some of the trends I believe PSS must follow and adopt to be successful in the next few decades.

Geographic data sharing and systems interoperability

Efforts to standardize the way in which we describe geographic features are critical to our ability to share government data between different departments, levels of government, and commercial and educational institutions. For example, if all municipalities called parcels by the same name and used the same terminology—and meaning—for a parcel's attributes, the cost of regional planning and administrative operations would be greatly reduced. In Europe, the problem has been less acute as most data collection occurs at the federal level. Therefore, work in this area is mainly happening in North America, where there is a strong tradition of local independence from federal control. The U.S Federal Geographic Data Committee and ESRI have strong programs in place to promote a common description of the most basic data sets used in government.

Of equal importance is the ability to locate and ingest another party's data with little or no human intervention in the conversion process. This is systems interoperability. The OpenGIS Consortium's standards for geographic data encoding (Geography Markup Language), geographic data publishing (Web Feature Service), and map publishing (Web Mapping Service) are being well received in the industry and provide a foundation upon which my work depends.

e-Government and homeland security

In addition to these technology trends, there is a strong push at the administrative and policy levels of the U.S. government to promote interoperable systems. Emergency response to natural disasters has been cited for years as a reason to invest in interoperable GIS systems. This argument did not resonate well at the highest levels of government, but emergency response to terrorism has focused the attention of policy makers on improving our ability to quickly gather and analyze geographic information. This new prioritization is seen in the Office of Management and Budget's E-Government Act of 2002, which lists GIS interoperability as one of ten key best practices funding priorities for federal agencies.

XML

The most important technology since the advent of the Web is Extensible Markup Language, or XML. XML is really nothing by itself. It is simply a framework in which to write languages for data encoding and system-to-system messaging. What XML provides is a consistent language structure and a way of describing the language's content in the form of XML Schema. Because structure and content are described in XML syntax, the software industry has built powerful, reliable tools to read XML on every operating system and application in common use. It is also very important that XML languages are plain text, so that their content is transparent to humans, even in the absence of computer programs that can read and manipulate the XML. This has a profound effect on people's trust in the content and in the ability of the content to be used in almost all current and future computing environments.

Web services

"Web services" is an umbrella term to describe systems that allow applications to communicate between computers using XML as a messaging language. The different communication implementation strategies go by many names (the most well known being SOAP, or Simple Object Access Protocol). However, the implementation strategies are not important in this context. What is most important is that all Web services strategies use a well-known and widely implemented Internet protocol for communication—HTTP—the foundation upon which all Web sites operate. While some technologists decry the drawbacks of the Web protocol, the advantages are numerous. The most obvious is that most organizations already have a Web infrastructure in place, so implementing Web services can be handled in a familiar way, and the wealth of Web software can be used to develop and run new Web service-based applications. The other important aspect of Web services is that they use XML for passing messages between computers, preserving the transparency that has made XML so popular and useful (although some implementations, most notably those promoted by Microsoft in their .NET framework, often still hide the actual message content (data) in a non-human readable format).

Identifying language development issues

Currently, interfaces to computing systems are mediated by people and their language, and usually by a hierarchy of people. In the terminology of economists, this results in a very high transaction cost for any change to the analytic process. This research seeks to reduce that transaction cost by creating a medium for the computer systems we have come to rely upon to interact directly with each other. This requires a language through which computers can "talk about" plans and models. What is the language of modeling? How can models be part of planning discourse when they are not written in a "language" that everyone understands? What theories drive the design of this language? This section provides the background in which these questions will be explored.

Choice of language environment

The first step in embarking upon this research area is to decide how to write the language. It is important to understand that software development or design is a secondary concern of this work. It only is addressed in regards to prototyping and proof-of-concept work. At its heart it is concerned with the design of a language to describe a process, specifically the planning analysis process, and formally tying it to computing methods.

My requirements for the language are as follows:

1. Human *and* computer readable
2. Operating system and application independent

3. Interoperable with World Wide Web and GIS standards
4. Clearly described and coupled with model execution

Based on these requirements, three possible implementations were identified: UML, XML and Java. Using a programming language such as Java, or even a scripting language like Perl or Python, fulfills many of the requirements, but code is only readable by programmers. This drawback could be partially alleviated by heavily documenting the code and using tools to present the documentation in narrative and graphic form, but I feel that this represents too close a tie between programming and modeling.

The UML (Unified Modeling Language) is an excellent candidate for this exercise because its graphic notation is accessible to a wider audience, while still retaining the features of a formal method. As stated by Muller (2000), "A method defines a reproducible path for obtaining reliable results. All knowledge-based activities use methods that vary in sophistication and formality. Cooks talk about recipes…architects use blueprints, and musicians follow rules of composition. Similarly, a software development method describes how to model and build software systems (Muller, 2000)." The UML method represents the software industry's consensus on how to graphically describe a software system. The primary problem with the UML is that a graphic notation is not computer readable. Also, the UML is independent of any execution environment, making it difficult to ensure that different applications can interpret the model in the same way and therefore interoperate. Software engineers use the UML to explain high-level ideas about system design, not to directly specify system execution.

While the UML may be used in this dissertation to help explain and document the language and software prototypes, the language itself will be written in XML. I feel that XML offers the best balance between the UML, which offers no implementation, and a programming language such as Java, which is inherently geared towards software engineers, and is too highly structured to offer the types of language elements necessary to construct a narrative. XML is a natural choice for three reasons. First, it offers the flexibility of full control over language definition, while remaining in a structured framework (XML Schema) that all systems can interpret. Second, it is becoming the dominant meta-language for network-aware, computer-to-computer communications, ensuring that many people will be familiar with its basic syntax, and will therefore shorten the learning curve for familiarizing themselves with the language. Finally, the major Web and GIS standards organizations use XML extensively, and there are a number of XML languages available to build upon that do useful things such as define database queries, encode geographic data and describe the Internet locations of resources.

A little more about XML

This research will use XML to develop a language called Planning Analysis and Modeling Markup Language (PAMML). XML stands for eXtensible Markup Language. It is a meta-language—a language designed for developing other languages. XML was developed as a way to tag information with metadata and enforce structural rules without requiring that the information be stored in or adhere to the strict rules of a database. It has proved to be a highly successful strategy, as the language is barely five years old and is already considered the standard for describing information outside of databases.

The benefit to writing a language in XML is that you can take advantage of a vast collection of software already developed to process XML, and only write the software that deals with the specifics of your particular language. Furthermore, one XML language can use others to describe generic entities. For example, XML language developers do not have to describe how a person's address should be written. They can simply use an XML address language developed by another information community (such as software companies that develop address book software). More importantly, a great deal of infrastructure needed to make an application work is common to all applications, such as security, authentication, data validation, etc. Using XML makes it possible for a language writer to be confident that their language can take advantage of advances in these areas without requiring major changes to their own work.

The way one develops an XML-based language is to write a rulebook. This is done in an XML language called XML Schema. This document functions as a dictionary—defining the set of terms that can be used—and also as a grammatical reference—enforcing rules about how words are put together to make sense. Additionally, XML Schema has the ability to reference other XML Schemas. This makes it possible to leverage existing work in related areas. PAMML will make extensive use of this mechanism to avoid re-inventing the wheel in the areas of networking, identity management, databases, and GIS. For example, whenever a link to an online resource is required, PAMML will use the World Wide Web Consortium's (W3C) XLink specification to identify the resource. Database access may take advantage of W3C's evolving XQuery specification. In the geographic field, a number of OpenGIS Consortium (OGC) specifications will be used. GML (Geography Markup Language) will be a supported data set format, and GML will also be used as the "native" geographic object language. WFS (Web Feature Service) will be a supported data format, in concert with the Filter encoding specification, which defines queries on geographic data.

Key Aspects of PAMML

Internet-centricity

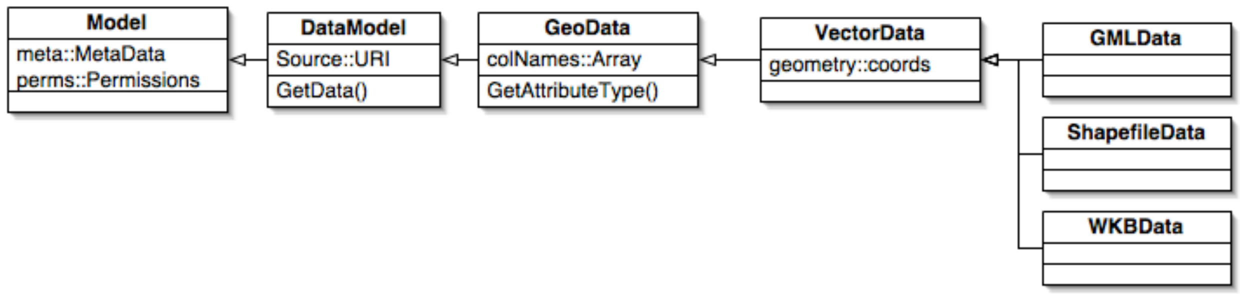
In this work, a model is an XML document that can be accessed via one or many URIs (Uniform Resource Identifiers, see http://www.w3.org/Addressing/ for a definition). URIs allow the model to be run, read, changed, or associated with an alternative model as described below in the section on controlling access. URI identification is critically important for these reasons:

- Local resources can be described in the same way as remote ones
- Model output can be a static file, or the result of a dynamic request, using standard Web technologies, e.g. CGI, Java Servlets or Active Server Pages
- Model publishing and execution can leverage Web server technologies for sharing and collaboration

Object-oriented design

Object-oriented (OO) is a term more often used with programming languages than data schema or language encoding, but many of the same ideas apply. OO has dominated programming language design for the last twenty years, beginning with Smalltalk and gaining widespread popularity with C++, Java, and most recently, C#. There are many benefits to OO design, but the most important to this effort is inheritance, which is the idea that an object can inherit functionality from another. This makes OO languages easier to understand, extend and program. For example, Figure 1 defines a GMLData (Geography Markup Language) data object. The reader of my model may have no idea what GML is, but because the GML object inherits from (is a child of) the abstract GeoData object, and the reader understands the properties of geographic data in general, most of the meaning about the properties of the GeoData object can be conveyed. More importantly, a model can be constructed that specifies a requirement for geographic data input via a GeoData object without being concerned about the actual data source, which may come from a Shapefile on disk (ShapefileData), a GML file (GMLData), or some other source.

Figure 1: Partial Geographic Data Object Hierarchy

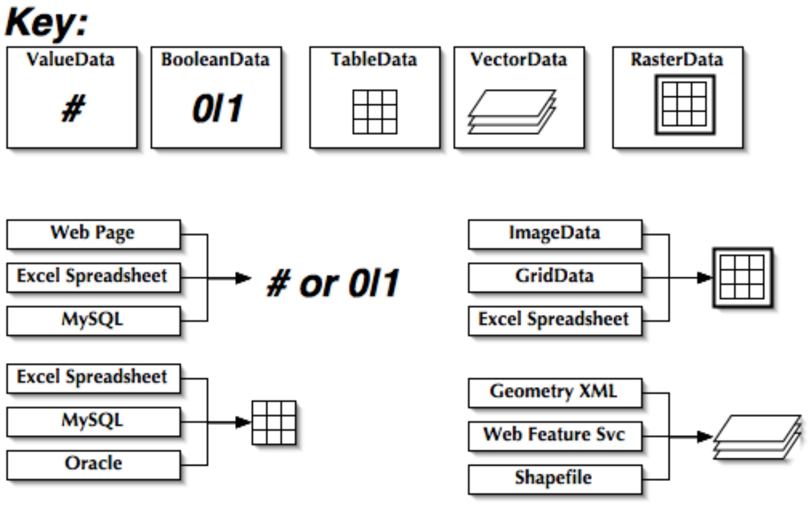


Design Patterns

Design patterns are ways to describe, at an abstract level, problems or situations that occur frequently. Christopher Alexander, speaking in an architectural context, describes their purpose saying, "Each pattern describes a problem which occurs over and over again in our environment, and then describes the core of a solution to that problem, in such a way that you can use this solution a million times over, without ever doing the same thing twice." The design pattern concept applies to decisions involved in designing the modeling language, but it also applies to the use of the language in practice. Patterns for solving planning problems can be captured, shared and applied in varied places and contexts.

The most pervasive pattern driving the encoding of the language is that components of a model are models themselves. This concept follows the **Composite design pattern**, which creates objects in such a way that different types of objects, as well as combinations of those objects, can be acted upon in the same way (actually, this language will have five types of data as shown in Figure 2, but this will not diminish the power of the composite design pattern). The importance of this design decision has implications throughout the creation and use of models. For example, every model has three basic features: data is input to the model; operations occur on that data; and data is an output from the model. This means that very complex models may be grouped together as one and described to a particular audience as a very simple, monolithic analysis. Yet, details are readily available to more technical audiences by examining the model's constituent parts. Another important benefit is extensibility. Even when the operations that occur are yet to be described by this proposed language, the analysis may still be included in, and useful to, a larger model that relies on it. For example, PAMML does not yet encode population growth forecasting models. However, the geographic data set output from the growth forecast can be used in a PAMML model.

Figure 2: Basic Data Types



What the language must express

At the computing level, the language must be able to describe basic spatial data and algorithms that are in common use. These are well defined after over thirty years of spatial information system use in planning. The nature of GIS is changing in response to the importance of the Internet, and this makes a host of new GIS paradigms important to capture. I take as a guide in this area the work of the OpenGIS Consortium, an international GIS standards organization that has encoded GIS software's fundamental properties into a series of documents called the "Abstract Specification." The language must also contain placeholders for concepts that are not currently be expressed well in software. These placeholders may look like traditional "black box" models, but at least they may be accommodated in the framework of a more open modeling language.

The next step in understanding language requirements is to understand the planning models that should be expressed. This is, of course, an infinite task, so I will focus my work on the most influential and widely used analytic model, site suitability analysis as first presented by Ian McHarg. Site suitability analysis includes the major elements of many planning analyses, from the basic manipulations of spatial data, to the assignments of weighted importance values based on subjective reasoning, to the importance of being able to substitute alternative data sets and sub-models when desired.

Assuring robust descriptive power using reflective prototyping

It is always difficult to define a methodology for proving this type of hypothesis. Traditionally, a methodology should involve coming up with a test that will either prove or disprove the original hypothesis. *My hypothesis is that modeling can be encoded into a formal language that supports collaborative analysis*. The most straightforward method of proving this hypothesis is to write the language and use it to model important collaborative analyses. This will be done, and will in fact be a major effort of the project.

As mentioned, the main research effort will be developing the modeling language, and prototype software that will perform the following modeling tasks:

- Develop models using a graphical user interface
- Develop models based on design patterns
- Access data
- Perform spatial operations on data
- Retrieve models and data over the Internet
- Visualize models graphically
- Visualize geographic results cartographically

It is important to note once again that PAMML is XML—simply a text file written in a specialized syntax. Therefore the creation of PAMML models does not require the use of specialized software. They can be created using any text editor. In this way it is just like HTML, which can be read and written in a text editor, but is much easier to understand when viewed in a Web browser.

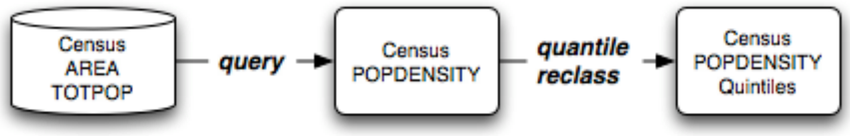
PAMML and its associated software will be evaluated by testing its ability to codify common analyses, like non-point source pollution modeling. This is an excellent case study for two reasons. First, there are a number of published models to estimate the amount of non-point source pollution in an urban area. Second, it involves a range of domain expertise, from understanding urban systems like land use and traffic, to natural systems like hydrology.

A demographic modeling scenario

Perhaps the best way to explain the full importance of this exercise is through an example scenario. Using Census data to understand demographics is one of the most basic planning analyses. Yet this seemingly simple task is executed with varying levels of quality, sophistication and depth of understanding. This scenario encodes a sophisticated analysis of population density into the PAMML and discusses how the model facilitates use and re-use of the technique.

Two analyses are presented in this example. In the first a very simple study is performed, calculating population density as the total population of a Census block group divided by its total area. Figure 3 diagrams the analysis. It shows raw Census data being queried to calculate a new attribute called POPDENSITY by dividing total population by area. Then a reclassification is performed to group the block groups into five classes, each having the same number of members—a quintile classification scheme.

Figure 3: Flow of a simple density analysis



Here are the same analysis steps presented in PAMML notation. For the first step, accessing Census 2000 population data for Cambridge, MA block groups, the PAMML might look like this:

Figure 4: PAMML for Census data

```
<ShapefileModel name="cambridgecensusbg"
   shpfile="file://camcen2kbg.shp"
   dbffile="file://camcen2kbg.dbf"
   shxfile="file://camcen2kbg.shx">
   <AttributeInfo>
      <Attribute name="totpop" datatype="xs:int" maxval="44444" minval="0"/>
      <Attribute name="area" datatype="xs:decimal" maxval="44444" minval="0"/>
   </AttributeInfo>
   <Meta>
      <Description>Cambridge, MA Census 2000 blockgroups</Description>
   </Meta>
</ShapefileModel>
```

There are two major components to the ShapefileModel. The most important is the location of the data, which is specified here with `shpfile`, `dbffile`, and `shxfile` attributes. The second is the listing of non-spatial attributes. These do not necessarily represent all the attributes present in the data, only those that the data model builder chooses to expose to others. The PAMML is a notation language, so it can not on its own prevent users from accessing all the attributes in the Shapefile, but one can imagine a data server that mediates between users and data, using the model's attribute list to enforce these rules. Finally, note that some Attributes have `minval` and `maxval` items. This is optional information that makes building models out of other models easier.

The second step is to calculate population density:

Figure 5: PAMML for query

```
<SpatialQueryVModel name="cambridgecensuspopdensity">
    <AttributeInfo>
        <Attribute name="uniqueid" datatype="xs:int"/>
        <Attribute name="popdensity" datatype="xs:decimal" query="totpop div area"/>
    </AttributeInfo>
    <Meta>
        <Description>Computes a population density attribute from Cambridge, MA Census 2000
blockgroups</Description>
    </Meta>
    <ShapefileModel name="cambridgecensusdata">
        <RemoteInfo uri="file://cambridgecensusbg.xml"/>
    </ShapefileModel>
</SpatialQueryVModel>
```

This model simply allows new attributes to be added to a spatial data set. The input is our ShapefileModel. It is specified via a reference to the first model, which contains the actual references to the Shapefile data. The output is a new spatial data set with two Attributes, **`uniqueid`**, which is copied from the ShapefileModel, and **`popdensity`**, which is the result of dividing the ShapefileModel's **`totpop`** Attribute by its area (notice that this model only uses the Attributes specified in the ShapefileModel). These two models could have been combined into one, but it is important that the most basic data model does not assume an execution environment more complex than the simple ability to read the data. The density calculation requires an execution environment that can do math, which is already outside the capabilities of standard Web servers.

The final step is to classify population density into meaningful groups. A quintile classification scheme is used, and the model looks like this:

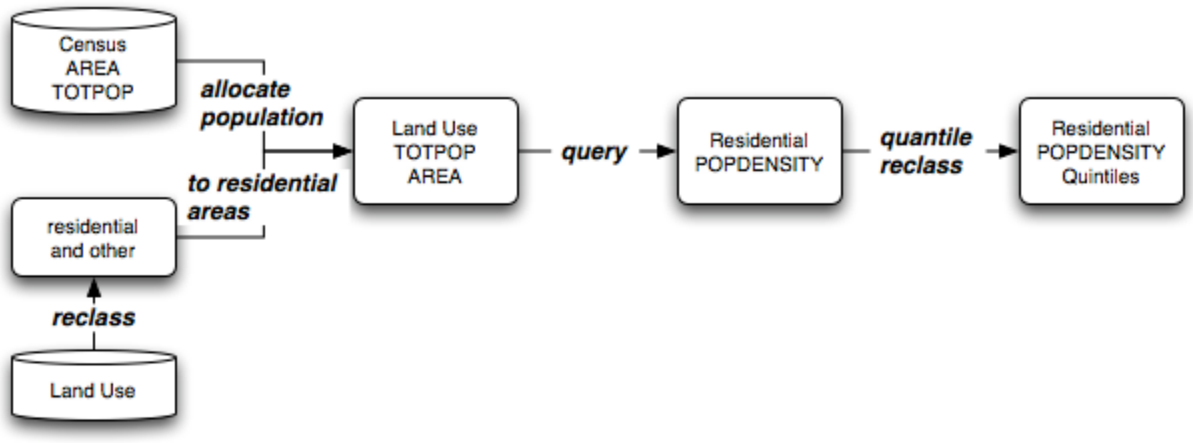
Figure 6: PAMML for quintile classification

```
<SpatialQuantileVModel name="Basic Population Density by quintile"
        usefeaturetype="popdensity"
        numranges="5">
    <AttributeInfo>
        <Attribute name="popdensityquintile" datatype="xs:decimal"/>
        <Attribute name="uniqueid" datatype="xs:int"/>
    </AttributeInfo>
    <Meta>
        <Description>Classified population density</Description>
    </Meta>
    <VectorDataModel name="cambridgecensuspopdensity">
        <RemoteInfo uri="file:///./cambma_popdensity.xml"/>
    </VectorDataModel>
</SpatialQuantileVModel>
```

The important parameters in this model are **`usefeaturetype`**, which specifies the Attribute to classify, and **`numranges`**, which specifies how many groups to create. Also notice that in this case a *VectorDataModel* is the input. A *SpatialQueryVModel* is a type of *VectorDataModel* (so is the *ShapefileModel*). There is no need here to know that the input data originated as a Shapefile, only that it represents spatial vector data, so the more general model type may be used.

Most planners with some amount of GIS training can perform the above analysis. It is not very good, however. Census analyses of this kind present a skewed portrait of where people live because it does not account for the fact that people do not live in office parks, forests, lakes, etc. A much better analysis would try to screen out these obvious biases. The next analysis uses land use data to constrain Census population counts to areas defined as residential. Notice in Figure 7 that the last three steps are the same as in the above analysis. The only difference is the input data set is residential land use with a `TOTPOP` attribute instead of the raw Census block groups.

Figure 7: Land use sensitive population density analysis



The model starts with a land use data set that is described similarly to the Census model shown in Figure 4. This is combined with a table describing land use codes that for brevity will not be shown here, but it specifies that 'R0', 'R1', and 'R2' are residential land use codes. The model in Figure 5 is used to group the land use data set into two categories, residential ('R') and other ('X'). The data is then "dissolved" using this attribute so that all adjacent polygons having the same land use code are merged. Note that this example nests a model "in-line," instead of referring to it via `RemoteInfo` notation.

After performing this task and then proportionally allocating Census population counts to residential areas (not shown in PAMML), the rest of the notation follows the first example, calculating a density attribute and classifying the values into quintiles.

Figure 8: PAMML for spatial reclassification and dissolve

```
<SpatialDissolveVModel name="cambridgeresidentiallanduse"
      usefeaturetype="res_lu">
   <AttributeInfo>
      <Attribute name="id" datatype="xs:int"/>
      <Attribute name="res_lu" datatype="xs:string"/>
      <Attribute name="area" datatype="xs:decimal"/>
   </AttributeInfo>
   <Meta>
      <Description>Residential land uses in Cambridge, MA</Description>
   </Meta>
   <SpatialReclassVModel name="cambridgeresluclasses">
      <AttributeInfo>
         <Attribute name="cambridgelanduse_id" datatype="xs:int"/>
         <Attribute name="res_lu" datatype="xs:string"/>
         <Attribute name="area" datatype="xs:decimal"/>
      </AttributeInfo>
      <Meta>
         <Description>Cambridge, MA 1:25000 land use</Description>
      </Meta>
      <ShapefileModel name="cambridgelanduse">
         <RemoteInfo uri="file:///cambma_lu.xml"/>
      </ShapefileModel>
      <ReclassTable name="landuse" joinfeature="lu21_code">
         <AttributeInfo>
            <Attribute name="lu21_code" datatype="xs:int"/>
            <Attribute name="residential" datatype="xs:string"/>
         </AttributeInfo>
         <table>
            <tr><att>R0</att><att>R</att></tr>
            <tr><att>R1</att><att>R</att></tr>
            <tr><att>R2</att><att>R</att></tr>
            <tr><att>*</att><att>X</att></tr>
         </table>
         <Meta>
            <Description>
               Changes land use to R (residential) and other (X)
            </Description>
         </Meta>
      </ReclassTable>
   </SpatialReclassVModel>
</SpatialDissolveVModel>
```

This may seem like an overwhelmingly complex description of a demographic analysis. In a sense, it is, and rightly so because it is a complex analysis, and a narrative description would be no shorter. This is, however, the wrong way to evaluate the language. In the early evolution of a machine language, the raw notation is usually written by hand, but as usage increases, tools are built to shield casual users from the complexity of the notation. In the second example, only five pieces of information are required from the user (and only the first two are needed for the simpler analysis presented previously):

1. The location of the Census data set
2. The Census total population and area fields
3. The location of the land use data set
4. The land use data's land use code field

5. The residential types of land use code

One can easily envision a graphical interface that makes obtaining this information from the user no more difficult than filling out forms on a Web page. Before these case-specific values are filled in, the model may be thought of as a template, or a design pattern. This term was used earlier in a software engineering context, where it helped to make decisions about how to design the language. It is also useful at a higher level, where we can think of spatial "design patterns" and planning "design patterns"—best practices models of how to extract certain kinds of knowledge from information.

Developing internal consistency with design patterns

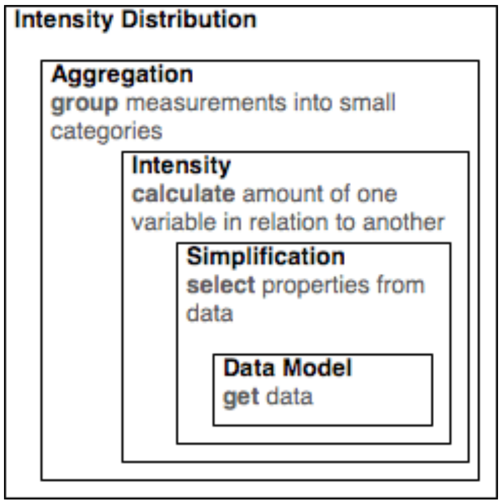
Design patterns are generic problem-solving models. They are the building blocks of analysis, like mathematical theorems. In fact, mathematical equations are the best-known formal design patterns. For example, calculus teaches us that rate of change can be determined by taking the first order derivative of the equation describing the phenomena. In statistics, we study the distribution of a sample by examining its variance and standard deviation. In the planning profession, we rarely use such formal design patterns, but when we do, their power is overwhelming. Zoning is a good example. The building rules articulated in a town's zoning code form a specific, and therefore very powerful pattern of how to develop land. Many planners feel that traditional Euclidean zoning is a flawed development strategy, but its rules continue to dominate the landscape. I would argue that the closer a planning theory gets to a design pattern, the more it is used in practice, mainly because planning theorists do not build cities, engineers, developers and elected officials do. And laws, which are simply a democratic definition of pattern-based rules, govern the activities of these people. The main purpose of this research is to show how PAMML is a tool to connect planning theory to analytic models and computational execution.

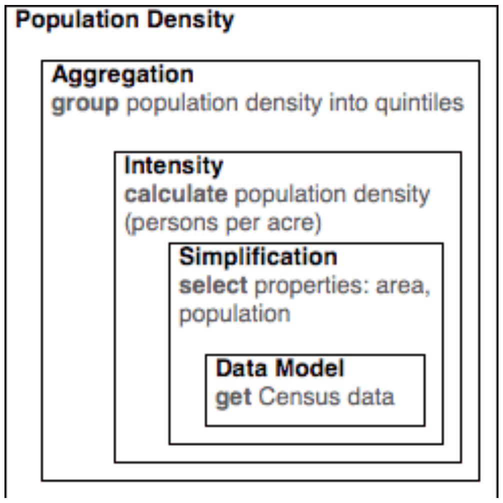
Planning design patterns currently are less formal. If you ask a planner how to site a new subdivision, a whole range of analyses will come to mind, involving land suitability, services provisioning, congestion, etc. While the solution to the subdivision siting problem is constrained somewhat by a person's professional education and practice (i.e. most planners can agree upon the validity of a number of subdivision siting methodologies), we do not build our analyses upon formal models, which are in turn constructed on a foundation of proofs and theorems.

The flow diagrams in Figure 3 and Figure 7 hint at an approach to presenting analysis from a pattern-based perspective, where the actions (lines) along with their inputs and outputs represent a unit of operation. But, by its nature as an XML Schema-based language, PAMML requires much more precision in its vocabulary. The two figures below illustrate the beginnings of a pattern language for generic spatial analysis (Figure 9), and a translation of that into a pattern language using the vocabulary of planning (Figure 10).

Figure 9:
Generic design pattern for analyzing intensity distributions

Figure 10:
Population density modeling as an intensity distribution design pattern


Intensity Distribution
Aggregation
group measurements into small categories
Intensity
calculate amount of one variable in relation to another
Simplification
select properties from data
Data Model
get data


Population Density
Aggregation
group population density into quintiles
Intensity
calculate population density (persons per acre)
Simplification
select properties: area, population
Data Model
get Census data

These analyses, obviously, are not novel, and neither is the idea of design patterns in planning, GIS, or computing. What is truly new is the ability to connect these powerful problem-solving modes from different disciplines in a way that enhances them all. The thesis will discuss in more depth how the design patterns facilitate more and higher quality analysis and drive intuitive user-interface design for the creation and presentation of PAMML-described models.

Goals

This research is an exploration into the future of geographic information systems for urban analysis, yet at the same time it is a return to proven, somewhat mundane analytic techniques. A look at the leading research journals show a field focused on studying the urban realm using highly sophisticated techniques like cellular automata and stochastic growth models. While these efforts are certainly important, they are best suited to an academic or federal audience at this stage in their maturity. My goal is to define a framework for analysis description and collaboration that can be extended to incorporate more complex modeling tasks in future work.

Planning

Guhathakurta (2002) says, "The circle of understanding and interpretation can only be completed through ordinary language." Only through discussion and debate do we as a society solve "wicked" problems. At first reading, this seems to be an argument against the use of XML or computer programming languages as a tool to promote understanding and interpretation. But Guhathakurta's statement is geared to society in general. My goal is to increase understanding among analytic professionals, such as planners, engineers and scientists, which allows me more leeway. He goes on to argue that analyses do not convince by virtue of their objective accuracy or "truth," but by how well they tell a story that resonates with their audience.

Some would argue that the main barrier to collaboration between experts is social. People are loath to share the details of their work for fear of their intelligence or expertise being called into question. While this situation certainly exists, I do not think it fully explains the lack of collaboration and information sharing we see today. A root cause may be that there is no way to present the work so that it resonates with others.

Guhathakurta's article suggests that *models should be written in a language that can be understood by a large cross-section of people*. This group most likely excludes the general public, but probably includes technologists in diverse fields of social and physical science. It also suggests that *a model should tell a story, reading more like a narrative than a computer program*. Beyond the software engineering and design choices that must drive this effort, these ideas will be used to guide the formulation of the modeling language.

Make modeling techniques and assumptions transparent through PAMML

My primary goal is to define a vocabulary for articulating the basic analytic processing operations. These building blocks of modeling fall into four categories: accessing data sources, re-categorizing data, mathematical operations (add, subtract, multiply, divide), and spatial operations (union, intersect, buffer). These operations will be described in an XML vocabulary that I am calling Analysis and Modeling Markup Language (PAMML).

A key feature of PAMML will be the ability to explicitly define assumptions that are subject to change and debate. For example, one operation may be to define a wetlands protection boundary. A purely mathematical definition might look like this in XML:

```
<BufferModel name=”Wetlands Protection Zone”>
   <BufferDistance units=”meters”>
      <Value>500</Value>
   </BufferDistance>
   <ShapefileDataModel name=”USFWS 1:12,000 Wetlands”>
      <DataSource>...</DataSource>
   </ShapefileDataModel>
</BufferModel >
```

PAMML will better articulate the societal priorities inherent in the value ‘500’ using vocabulary like this:

```
<BufferModel name="Wetlands Protection Zone">
   <BufferDistance units="meters">
      <Value>
         <ValueDataModel name="Buffer for healthy wetlands">
            <ValueData>
               <Value units="meters">500</Value>
               <Meta>
                  <Description>Based on a 1994 EPA
                     study: http://www.epa.gov/wetlandsReport.html
                  </Description>
               </Meta>
            </ValueData>
            <Meta>
               <Description>
                  Suggested distance residential development should
                  be set back from wetlands.
               </Description>
            </Meta>
         </ValueDataModel>
      </Value>
   </BufferDistance>
   <ShapefileDataModel name="USFWS 1:12,000 Wetlands">
      <DataSource>...</DataSource>
   </ShapefileDataModel>
</BufferModel>
```

The difference in the actual language is subtle, but the decision to make a numerical value a model in its own right allows key metadata to be preserved, and begins to construct a narrative around the analytic process.

Connect models across networks

PAMML will also support the ability to use a model component located elsewhere on the Internet. This example retrieves the wetland's buffer distance from another PAMML file at an imagined remote research center. Using standard Web technology, the model can be an XML file, or a program whose output is a number, and its execution is triggered by a request for information. This allows the program to compute its output based on the most current available data. Note also the *CacheInfo* element. This tells the user that the number in the *Value* element has been retrieved from the URI listed in the *RemoteInfo* element. The user may choose to use the cached value or request that the value be updated. This flexibility is extremely useful in situations where a connection to the remote resource is unavailable, or the model is being run repeatedly in a short time frame when the value is unlikely to change.

```
<BufferModel name="Wetlands Protection Zone">
    <BufferDistance units="meters">
        <Value>
            <ValueDataModel name="Buffer for healthy wetlands">
                <ValueData>
                    <Value units="meters">500</Value>
                    <Meta>
                        <Description>Based on a 1994 EPA
                            study: http://www.epa.gov/wetlandsReport.html
                        </Description>
                    </Meta>
                    <RemoteInfo name="Wetlands buffer distance"
                        uri="http://web.mit.edu/buffermodel.xml@distance"/>
                    <CacheInfo cached="true"
                        time="2002-07-12T10:35:58-05:00"/>
                </ValueData>
        ...
</BufferModel>
```

Allow multiple users to create alternative scenarios and alter assumptions

This example shows how the language offers the ability to experiment with and manage alternative scenarios. This *ValueDataModel* contains multiple *ValueData* elements, conveying the meaning that the *Value*s may be treated as alternatives to each other. A *rating* attribute is added to articulate the model creator's opinion of which value is "better." The model's user may, of course, ignore the rating, but by specifying multiple values for the same variable, we capture conflicting viewpoints, and hint at how PAMML can begin to describe the debate around a contentious issue.

```
<BufferModel name="Wetlands Protection Zone">
    <BufferDistance units="meters">
        <Value>
            <ValueDataModel name="Buffer for healthy wetlands">
                <ValueData rating="1">
                    <Value units="meters">500</Value>
                    <Meta>
                        <Description>Based on a 1994 EPA
                            study: http://www.epa.gov/wetlandsReport.html
                        </Description>
                    </Meta>
                </ValueData>
                <ValueData rating="3">
                    <Value units="miles">0.5</Value>
                    <Meta>
                        <Description>Based on an undergraduate landscape
                                architecture thesis written in 1999:
                                http://www.umass.edu/people/jhenry/thesis.html
                        </Description>
                    </Meta>
                </ValueData>
                <Meta>
                    <Description>
                        Suggested distance residential development should
                        be set back from wetlands.
                    </Description>
                </Meta>
            </ValueDataModel>
        </Value>
    </BufferDistance>
    <ShapefileDataModel name="USFWS 1:12,000 Wetlands">
        <DataSource>...</DataSource>
    </ShapefileDataModel>
</BufferModel>
```

These examples offer a flavor of the language that will be articulated, but they are very preliminary, and only intended to provide the reader with a gentle introduction to the work.

Evaluation Measures

Evaluating success in this endeavor is a difficult task. Many aspects of success are highly subjective—many languages have gained widespread acceptance even though they are poorly designed from a computer science perspective and have limited functionality. However, there are some measures by which the language's fitness for use can be evaluated.

Robustness is a basic performance measure. PAMML must be able to describe the most common data formats, algorithms and functions in spatial analysis. It must also be able to model common planning analyses such as thematic mapping and McHarg-ian overlay analysis. The next most important measure is **interoperability**. As this work is largely premised on an environment where work is distributed across networked computers with heterogeneous software environments, it is crucial that PAMML interoperate with standard protocols such as TCP/IP, HTTP and SOAP, as well as messaging standards like the OpenGIS interfaces (WMS, WFS, SensorML, etc.) and the

World Wide Web consortium's stack of XML standards (XML, XML Schema XPath, XQuery, SOAP, etc.). In order for PAMML to define an analysis to be executed by a computer, it must not only be robust (in that it has the descriptive power to detail all the steps). It must also be unambiguous, leaving no room for misinterpretation of what decisions to make. This is measure is called **specificity**. Finally, PAMML should have **extensibility**. The basic notation will be very engineering oriented, but most users will want to define a higher-level notation that more closely resembles the "terms of art" used in their profession. For example, a landscape architect and an urban designer may perform analyses that use the same underlying spatial functions, but have different names in their respective professions. PAMML should be able to accommodate the ability to define higher-level notation languages that shield users from the engineering terms, or aggregate a group of small analytic tasks into a larger one.

LITERATURE REVIEW

Reflecting on the science of GIS

"How do we explain geographical phenomena through the application of appropriate methods of analysis, and models of physical and human processes? Under what circumstances is the scientist willing to trust data that he or she did not collect, and will the increased technological ability to share scientific data over the Internet…change them? Such questions about tools often have their roots in theoretical questions about appropriate representations, operations, and concepts."

Goodchild, et al., IJGIS 1999

These fundamental questions are posed in a 1999 article co-authored by many of the elder states-people of the field, including Mike Goodchild, Max Egenhofer and Karen Kemp. One might suppose that thirty years into the evolution of GIS these issues would have been discussed in great depth. Yet the article introduces an NSF-funded effort, Project Varenius, which seeks to build the theoretical foundation of geographic information sciences that was neglected during decades of practice-oriented work.

My project, while concretely grounded in a prototype implementation, fits well into the research agenda expressed by Project Varenius. It provides one answer to Goodchild's question. I do not hope to provide the definitive answer—it will take years of work by a community of researchers—my main goal is to encourage the field to step back and work on these fundamental, broad-based problems that are still studied too little.

This research effort takes place in, and is in many ways motivated by, recent activities in a number of areas. First and foremost is a sense that the field of geographic information systems is ripe for a period of reflection on the theoretical foundations of the discipline after a number of decades of successful practical applications.

Place-based modeling

The primary literature consulted for this work is in the field of geographic, or place-based modeling. I consider modern work in this area to have started with Ian McHarg's

(1969) seminal book, *Design with Nature*, where he lays out the technique of map overlay analysis. Despite, or maybe due to, its simplicity, this is still the most common analytic technique for land-based analysis and planning. In fact, the focus of PAMML in this first incarnation is to support McHarg-ian style map overlay analysis.

Geographic modeling is a very large field, and attempting to undertake an exhaustive review of the modeling literature would be too big an effort and only lead to confusion. However, a general background in the types of models is appropriate to have, and Alberti (1999) provides an excellent review, describing a wide range of modeling techniques used in the last forty years. The major categories are those descended from the Lowry gravity model; economic market-based models such as California Urban Futures 2; and micro-simulations, such as UrbanSim and Clarke's (1997) urban growth model. For my purposes, what is important in these models is that they are rarely self-contained, but rely on endogenous variables and models from other disciplines, like transportation and economics. Also, they usually make use of advanced statistical techniques like logistic regression. The prevailing trend in modeling in the last decade or so has been to more accurately simulate change by using Monte Carlo approaches, allowing future change to be based on previous actions. Finally, the latest trend has been towards using cellular automata, or agent-based modeling to focus the process on understanding urban processes from the point of view of actors in a changing economic and geographic landscape, and to understand macro-phenomena as the accumulated impact of simple, autonomous micro-behavior. I envision this work being extended to support these more complex urban models in the future, but that is not the focus of this dissertation.

Non-point source pollution modeling

For an understanding of the models used for my test case, non-point source pollution modeling, I will rely heavily on Marsh (1991). He provides a highly practical handbook on how to construct and use these types of models. I also use reports from the NEMO (Nonpoint Education for Municipal Officials) project at the University of Connecticut.

Indicators and performance measurement efforts

I have hinted that one implication of this work is its ability to transform the goals of analysis from report generation to situation monitoring and performance measurement. The need to support this effort is evident in many places. The National Neighborhood Indicators Partnership is an effort to build "advanced information systems with integrated and recurrently updated information on neighborhood conditions in their cities (http://www.urban.org/nnip/concept.html)." This is the most explicit example of this change in focus, but the trend presents itself in many other places, like the Heinz Center's *Report on the State of the Nation's Ecosystems*(2002), which recommends that environmental quality be monitored and reported in a consistent, constant way, in the manner of well-known federal economic indicators such as durable goods orders, housing production, consumer spending, etc. Indirectly related efforts include local government efforts to define a strategy for integrating the Internet into their mission. The National Civic League addresses this in the 8th revision of their Model City Charter. A joint project of the

National Association of Counties and the National League of Cities seeks to support the ability of towns to automate government transactions over the Web through their "Totally Web Government" program. Literature in this area will help define requirements for the collaborative aspects of the language.

The changing nature of data

Surprisingly, encoding and accessing geographic data is still an active research area, and it is highly relevant to this project. There are two problem areas in geographic information today. First, governments use different terminology to describe the same things, such as zoning categories, property characteristics, and even natural features. The Federal Geographic Data Committee's National Spatial Data Infrastructure Initiative (NSDI) seeks to promote a framework for consistently describing basic geographic data sets, in conjunction with local agencies and ESRI, the primary GIS software provider for municipalities (see http://www.fgdc.gov/framework/framework.html and http://www.esri.com/software/arcgisdatamodels/index.html).The American Planning Association is promoting land description standard that succeeds the venerable Anderson land use classification scheme. This standard includes activities as well as land cover (http://www.planning.org/LBCS/GeneralInfo/).

SIGNIFICANCE

Richard Klosterman writes pessimistically in the anniversary issue of *Environment and Planning B* (Klosterman 1998), "Ten years ago John Landis and I suggested in this journal that the continued rapid advances in microcomputer technology could lead to the development of…small modules which perform analytical tasks and are linked to other modules and to common data sets. These modules will operate at different geographic levels, include a range of different analytical tasks, provide interactive, geographically referenced data and graphic input and output, and be capable of easily transferring data into and out of public and private data banks, and popular spreadsheet, database, CADD and GIS packages. Regrettably, these tools are no more available now than they were ten years ago." He goes on to state that academics will continue to develop interesting prototypes and that the commercial GIS industry will continue to build software for focused problems such as permit tracking and transport planning.

In 1998, I would have agreed completely with Klosterman's assessment of the planning support systems field. In 2004, however, important structural changes have occurred. Web services technology provides the framework in which to build "small modules which perform analytical tasks and are linked to other modules and data sets." The OpenGIS consortium's success in garnering broad industry support for interoperable data exchange and encoding standards makes it possible for the work of academics to be integrated into commercial software implementations. And the use of XML as an encoding and messaging language allows spreadsheet, database, CADD and GIS packages to communicate with each other.

GIS Infrastructure

I have already intimated at the importance of having a vision for the future of government information systems to guide agencies in making critical investment decisions in human capital, computing infrastructure, GIS policy in the next ten to fifteen years. I argue that this work, in concert with a vision for semantic data encoding, is a powerful and robust solution.

Expert Collaboration

A recurring theme in this research will be the lack of horizontal (e.g. assessing, planning, public works agencies) and vertical (local, state, federal) collaboration in planning. The ideas expressed here create the environment for collaboration to become a reality by clearly defining "who is doing what to whom, and when." While this work does not seek to improve analytic methodologies, it presents the current methods clearly for the first time.

Design Patterns

One significant implication of this work that has barely been mentioned is the potential to create, articulate and implement design patterns using PAMML. The urban design field, especially, provides us with a rich history of design patterns that are beautifully articulated (in English), but are difficult to apply outside the designer's original context. As far back as Olmstead's emerald necklaces of the 19th century, that designer was able to implement his vision of how to integrate green space into a metropolis, but despite the popularity of the idea, there have been few independent emerald necklaces in the world. The same can be said of Kevin Lynch's "good city form," Christopher Alexander's "pattern language," or Alan Jacobs' "great streets." The argument can be applied to the more popular urban models in recent times, such as Landis' California Urban Futures or Bill Hillier's Space Syntax. The expertise of these models must be captured in a way that allows their rules to be applied in different scenarios and physical contexts.

Paradigm Shift

More than anything else, this work aims to change the paradigm of analysis from the current one-time major effort to produce a document to many small efforts that produce a continuous information flow. As Lew Hopkins states, "Making plans for urban development is something you do constantly, not once" (Hopkins, 1999). The underlying assumptions that go into a plan, such as economic conditions and development activity constantly change, yet most plans are static. This is a necessary compromise based on the cost of marshalling the resources required to prepare useful plans. The framework suggested here allows plans to become dynamic tools—more like monitoring and early warning instruments than traditional plans. This may sound threatening to those who consider plans to be embodiments of a community's vision about their place, but Hopkins notes that plans are really the strategic implementation of visions, not the visions themselves. In this new type of plan, the community's vision is still present. It simply manifests itself in a different form, such as the point at which development triggers a moratorium or an infrastructure investment.

RESOURCE ALLOCATION

Timetable

Task	Jan	Feb	Mar	Apr	May	Jun	Jul	Aug	Sep	Oct
Additional literature review		■	■	■	■					
Refine thought experiment			■	■						
System conceptual design			■	■						
Overall system software design and specifications		■	■	■						
Specification of XML encodings		■	■	■						
Programming		■	■	■	■					
Formulate exemplar modeling tasks			■	■						
Test the system on the example tasks				■	■					
Empirical study with students					■	■				
Theoretical and practical implications						■	■			
Write the full draft							■	■	■	
Final draft								■	■	■

Software Resources

Building prototype software that uses the proposed PAMML is a major investment of time. This effort is only practical by leveraging the software base of a number of open source software projects. All code will be written in Java, which is compatible with all of the following tools:

JGraph (http://jgraph.sourceforge.net/): JGraph's mission is to "provide a freely available, standards-compliant and thoroughly documented open source component to display and edit graphs (networks) with Java." JGraph is used as the graphical user interface for building PAMML model components. PAMML is not dependent on this, however, as a user can always write PAMML in any text editor or invent their own editing environment.

GeoTools (http://www.geotools.org/): GeoTools is a Java toolkit for performing many basic GIS functions, such as reading data, converting it into other coordinate systems and projections, and visualizing with maps.

GeoServer (http://geoserver.sourceforge.net): This is an implementation of the OGC Web Feature Service specification, offering a modifiable, freely distributable geographic feature server.

Java Topology Suite (http://www.vividsolutions.com/jts/jtshome.htm): This is a Java API of 2D spatial predicates and functions. It provides spatial analysis methods such as buffer, intersection, etc. JTS is an open source library whose development has been funded by GeoConnections, a national partnership working to build the Canadian Geospatial Data Infrastructure.

Standards

As much as possible, this work will leverage existing OpenGIS Consortium specifications to promote interoperability with existing GIS systems and World Wide Web Consortium standards for Web interoperability.

OpenGIS Consortium standards

Web Mapping Server Specification (WMS), version 1.1.1, http://www.opengis.org/techno/specs/01-068r3.pdf.

Style Layer Descriptor Specification (SLD), version 1.0, http://www.opengis.org/techno/specs/02-070.pdf.

Web Feature Server Specification (WFS), version 1.0, http://www.opengis.org/techno/specs/02-058.pdf.

Geography Markup Language (GML) version 2.1.2, http://www.opengis.net/gml/02-069/GML2-12.pdf.

Web Coverage Server Specification (WCS), discussion paper, http://www.opengis.org/techno/discussions/02-024.pdf.

Simple Features Specification for SQL, version 1.1, http://www.opengis.org/techno/specs/99-049.pdf.

World Wide Web Consortium standards

XML. http://www.w3.org/XML.

XML Schema, version 1.0, http://www.w3.org/XML/Schema.

XML Query Language, version 1.0 and XPath version 2.0, http://www.w3.org/XML/Query.

XML Linking Language (XLink), version 1.0, http://www.w3.org/TR/xlink.

Web Services Definition Language (WSDL), version 1.1, http://www.w3.org/TR/wsdl.

BIBLIOGRAPHY

Alberti, M. 1999. "Modeling the urban ecosystem: a conceptual framework" *Environment and Planning B: Planning & Design* **26** 605-630.

Alexander, C., Ishikawa, S. and Silverstein, M. 1977. *A pattern language: towns, buildings and construction*.

Guhathakurta, Subhrajit. 2002. "Urban modeling as storytelling: using simulation models as a narrative" *Environment and Planning B: Planning & Design* **29** 895-911.

Heinz, Center for Science, Economics and the Environment. 2002. *The state of the nation's ecosystems: measuring the lands, waters and living resources of the United States*.

Hopkins, Lew. 2001. *Urban Development: the logic of making plans*.

Jacobs, Allan. 1993. *Great Streets*.

Klosterman, Richard 1998. "Computer applications in planning" *Environment and Planning B: Planning & Design* 32-36.

Lee, Douglass B. 1973. "Requiem for large scale models" *Journal of the American Institute of Planners* **39** 163-178.

Lynch, Kevin. 1960. *The Image of the City*.

Lynch, Kevin. 1981. *Good City Form*.

McHarg, Ian L. 1969. *Design with Nature*.

Muller, P. 2000. *Instant UML*.

OpenGIS Consortium. 1999. *Abstract Specification, Topic 5: The OpenGIS Feature*, http://www.opengis.org/techno/abstract/99-105r2.pdf.

OpenGIS Consortium. 2000. *Abstract Specification, Topic 6: The Coverage Type*, http://www.opengis.org/techno/abstract/00-106.pdf.

OpenGIS Consortium. 2002. *Abstract Specification, Topic 12: The OpenGIS Service Architecture*, http://www.opengis.org/techno/abstract/02-112.pdf.

OpenGIS Consortium. 2001. *OpenGIS® Geography Markup Language (GML) Implementation Specification*, http://www.opengis.net/gml/01-029/GML2.pdf.

OpenGIS Consortium. 2002. *OpenGIS® Web Feature Service Implementation Specification*, http://www.opengis.org/techno/specs/02-058.pdf.

Steinitz, Carl. 1995. "Design is a verb; Design is a noun" *Landscape Journal*.