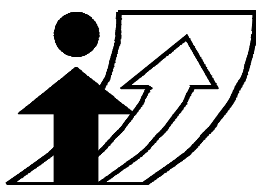# ILUTE: An Operational Prototype of a Comprehensive Microsimulation Model of Urban Systems

Paul Salvini, University of Toronto
Eric J. Miller, University of Toronto

Conference paper
Session IV: Microsimulation

ILUTE: An Operational Prototype of a Comprehensive Microsimulation Model of Urban Systems

Paul Salvini and Eric J. Miller
Department of Civil Engineering
University of Toronto
Toronto, Canada

Phone: 416-686-0221
eMail: psalvini@alumni.uwaterloo.ca, miller@civ.utoronto.ca

Abstract

This paper describes the development of an operational prototype for a comprehensive microsimulation model of urban systems. It examines several important design advances that emerged during the transition from a conceptual framework to operational code.

ILUTE (Integrated Land Use, Transportation, Environment) simulates the evolution of an integrated urban system over an extended period of time. This model is intended to replace conventional models for the analysis of a broad range of transportation, housing and other urban policies.

An overview of the ILUTE framework was presented at the 9th IATBR conference (Miller and Salvini, 2001). Since then, considerable progress has been made on the overall model and its component submodels. At present, an operational prototype is being tested using data from the Greater Toronto Area. Disaggregate information for the model is synthesized from census data, travel survey data, activity data, and randomly generated proxy data.

ILUTE is based on the "ideal model" described in the final report of the Transit Cooperative Research Program's (TCRP) Project H-12, "Integrated Urban Models for Simulation of Transit and Land-Use Policies" (Miller, Kriger, and Hunt, 1998).

Keywords

Microsimulation, ILUTE, Urban Systems, Modelling, International Conference on Travel Behaviour Research, IATBR, Transportation and Land Use

Preferred citation

Salvini, Paul and Eric J. Miller (2003) ILUTE: An Operational Prototype of a Comprehensive Microsimulation Model of Urban Systems, paper presented at the 10th International Conference on Travel Behaviour Research, Lucerne, August 2003.

1. Introduction to ILUTE

ILUTE simulates the activities of individual objects (agents) as they evolve over time. These objects include persons (with households and families), transportation networks (road and transit networks with support for bicycle and walking modes), the built environment (houses and commercial buildings), firms, the economy (interest rates and inflation), and the job market. The simulator evolves the state of the urban system from a specified base month to a specified target month. At any time, the simulation can be branched to test various policy alternatives. Figure 1 provides a structural overview of the ILUTE model as outlined in Miller, Kriger, and Hunt (1998).

Figure 1   Structural Overview of the ILUTE model
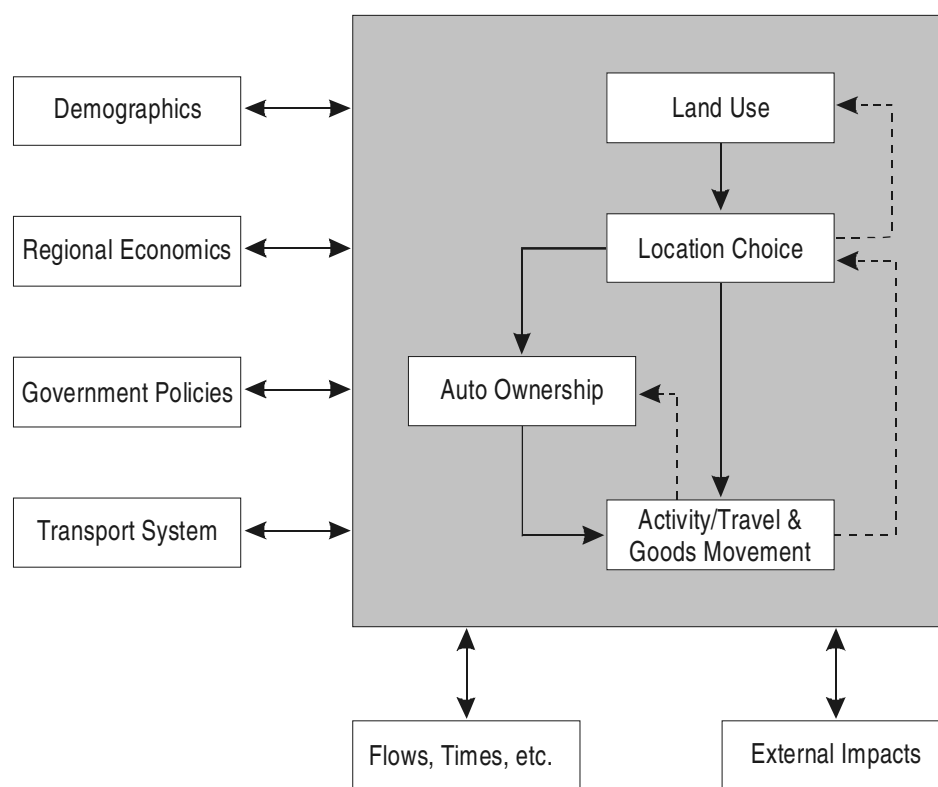
Source: Miller, Kriger, Hunt (1998)

The shaded "behavioural core" of the model has four inter-related components: land use, location choice, auto ownership, and activity / travel. ILUTE enables researchers to capture the complex interactions that occur within an urban system. The transportation system, for example, is one of many interconnected factors affecting the quality of "life" within the simulated system.

As an integrated full-feedback model, ILUTE allows higher-level decisions (e.g. residential mobility) to influence lower-level decisions (e.g. daily travel behaviour) and vice versa. A variety of modelling methods are employed within ILUTE to capture object behaviours: state transition models, random utility models, rule-based (computational) models, learning models, exploration models, and newly developed hybrids of these approaches.

ILUTE supports several output options: the system state can be exported as a set of binary files (for efficient storage and retrieval) or as relational database tables (for easy import into other packages). The resulting spatiotemporal data can be exported for visualization using the Houdini 3D animation tools from Side Effects Software.

1.1   Current Status of the ILUTE Operational Prototype

The ILUTE prototype consists of approximately 15,000 lines (over 200 pages) of C++ source code in approximately 50 classes. The code supports the following capabilities:

- synthesize a test set of households, persons, jobs, dwelling units, and buildings

- import spatial data (census tracts, Transportation Tomorrow Survey (TTS) planning districts, TTS traffic zones), transit and road networks, EMME/2 travel time data (by mode and time of day), and text-based economic data (interest rates and consumer price indices)

- evolve the state of the system to an arbitrary date using an arbitrary time step by simulating the activities and behaviours of individual objects (e.g. persons and households)

- track (display) the activities and behaviours of individual objects in the system as they evolve

- simulate population in-migration and out-migration

- export spatiotemporal data for visualization in 3D (static or animated)

- read and write state information to a relational database

1.2   Advances of the ILUTE Operational Prototype

The ILUTE operational prototype presents several advances to the state-of-the-art:

1.   ILUTE is designed as an open test bed that can accommodate the work of dozens of researchers. Given the complexity of urban systems, this approach is essential to building a credible next-generation model. Through the use of individually accessible submodels, researchers are able to work independently, yet contribute to a common goal.

2.   ILUTE possesses an object-oriented design and a class structure that accommodates the demands of the "ideal model". The design contains a rich set of real-world objects with appropriate semantics (attributes and operations) and relationships (generalization, composition, aggregation, etc.).

3.   ILUTE explicitly represents households, persons, and families. The relationships between these entities are created during synthesis and maintained during evolution.

4.   Support for multiple spatial aggregations allows rich submodels to be developed. A clear example of the benefit of multiple spatial representations is developed in the residential search process where the housing market submodel is able to use two levels of spatial aggregation: the first to narrow down the district, and the second to narrow down the neighbourhood.

5.   ILUTE's ability to handle joint decisions through ad-hoc decision making units provides an important extension to the benefits of explicitly representing persons, households, and families. While many decisions are made at the person and household level, other decisions are made jointly by persons who do not share the same household (or who represent only a portion of a household).

6.   The ability to handle temporal leads and lags enables ILUTE to better represent real-world decisions. While it was initially thought that leads and lags could be handled at subsequent (or previous) time periods, doing so results in decisions being made outside of their correct economic context.

7.   The stressor mechanism is an innovative and conceptually appealing approach to handling triggered events, joint decisions, accumulated stress, and person-household interaction. The behavioural capabilities of this mechanism enrich the behavioural credibility of ILUTE. Authors of future submodels can use the stressor mechanism as an elegant way to simplify the handling of complex and nested decision-making processes.

8.   ILUTE's time stamping mechanism allows a variable time step to be used throughout the model. This flexibility improves computational efficiency and model accuracy over fixed-time approaches. The support of a variable time increment allows both long-run and short-run processes (and everything in between) to be handled at their optimal temporal frequency.

9.   The ILUTE prototype implements a flexible and extensible mechanism for loading external temporal data. The temporal data manager in the ILUTE prototype avoids many common data handling errors by providing a common interface for loading and storing raw data samples and interpolating at run time.

10.   ILUTE's monetary value handling method of storing a date-value pair offers significant conceptual and practical advantages over traditional approaches. The auto transactions submodel currently implemented within the ILUTE prototype, for example, makes use of the monetary conversion mechanism to generate income information specific to a calibrated year.

11.   A representative "stub" model of the housing market highlights several of ILUTE's architectural strengths. The multi-stage process of activity, search, and bid is implemented in the submodel and the process is triggered either as a random event or as the result of accumulated household stress exceeding a prescribed threshold.

2.   Design of the ILUTE Operational Prototype

Early in the development of ILUTE, it was clear that the object-oriented approach to software development would have a profound impact on the conceptual appeal of the overall model. The one-to-one mapping between objects in the "real world" and those in the microsimulation model enables researchers to describe behaviours in the language of the problem domain.

2.1   Representation of Persons, Households, and Families

One of the key features of the ILUTE operational prototype is the explicit representation of persons, households, and families. While this requirement adds considerable complexity to the model, it is essential to adequately reflect the context in which real-world decisions occur. The inclusion of the family relationship within ILUTE enables a number of insightful decisions to be made in location choice and activity scheduling. Family relationships captured within the ILUTE operational prototype include mother, father, spouse, ex-spouses, children, and siblings.

2.2   Behaviour and Decision-Making

In ILUTE, behaviour is implemented by objects called *decision-making units*. ILUTE's explicit representation of compound groups like households and families means that decisions can be made directly by these objects. In other cases, several objects will collaborate to form ad-hoc decision-making units (for example, the drivers in a household may collaborate to make a decision about a vehicle purchase). The assignment or attribution of behaviours to collective decision-making units is a fundamental design feature of the ILUTE model. This feature delivers several practical benefits, including the ability to trace decisions on a per-entity basis and the ability to support multiple temporal decision scales. Some decision-making units are abstractions of real-world entities. A firm, for example, is an object within ILUTE that makes decisions. While in the real-world, it is persons in the firm that are actually making these decisions, the use of such objects as surrogate decision-makers is a useful and understandable abstraction of the real-world process.

2.3   State Representation

An object's knowledge about itself and its environment is stored within each object in attributes that form the object's state. Any past memory or knowledge required by the objects in the model are carried forward to the present time. One of the challenges in developing a large-scale microsimulation model is handling the fact that each person has a unique representation of reality. For example, we each have a mental map of the city in which we live. Someone who lives in Toronto might have a good idea of the road network in Toronto but would probably have a poor representation of the road network in Lucerne. While it is convenient to think of all persons as sharing a single perfect map of the city, we know that this perfect representation is far from accurate. In order to simulate the absence of perfect information, the true system state is stored once and individual accesses to that state are artificially adjusted during the search process to yield imperfect results.

2.4   Spatiotemporal Data

The evolution of an urban area involves activities that take place in space and time. As both space and time are *continuous* variables in the real world, some form of discretization is necessary in order to store these values in a simulation.

In discretizing time, it was deemed advantageous to accommodate multiple temporal resolutions. The design involves storing a timestamp along with every object and update

process in the system. This timestamp allows objects to be updated and processes to be executed on a flexible schedule. The creation of a separate timestamp class enables the basic temporal resolution of the simulation to be easily modified. The flexibility of variable time increments is unique to ILUTE and serves to significantly improve computational efficiency and model accuracy over fixed-time approaches. The fixed one-year time step suggested in the description of the ideal model was a compromise between a one-month and a multi-year time step. For some events, such as simulating the housing market, one year is probably too infrequent. For other events, such as simulating transportation infrastructure changes, one year is probably too frequent. The support of a variable time increment allows all processes and objects to be updated at their optimal temporal frequency.

To handle the representation of space, the ILUTE operational prototype also accommodates multiple spatial resolutions. Census tracts, planning districts, travel zones, and grid squares are all handled simultaneously in the ILUTE prototype. The flexibility to handle multiple spatial aggregations (with any underlying spatial representation) is a key feature of the model.

3.   Architecture: Class Structure

The list of classes in Table 1 provides some insight into the current architecture of ILUTE:

Table 1   Application Class Diagram - Operations

ActivityGenerator	Application	AutoTransactionModel
Bid	Building	CDBHousehold
CDBPerson	CDBPersonChildList	CDBPersonExSpseList
CDBPersonSiblingList	CDBPropertyOwner	CRecordset
DatabaseAdministrator	DwellingUnit	Establishment
Firm	Household	HousingMarket
Job	Link	Location
MarriageMarketPool	MarriageMktModerator	Matrix
MonetaryValue	Neighbourhood	Node
Person	Polygon2D	PropertyOwner
RoadLink	RoadNode	Schedule
SHP_Polygon	SimulatedObject	SimulationDate
SpatialObject	StressManager	Stressor
StressorFinancial	StressorTravelTime	TemporalDataManager
TransitLink	TransitNode	TransportationNetwork
TravelTimes	TTSPlanningDistrict	Vehicle
Vertex2D	World	

3.1   The Application Class

At the highest level, ILUTE consists of an Application class that serves as the master controller for the simulation. The Application class manages a simulated world and controls all user interaction. The Application class has been separated from the World class in order to allow the application to load multiple worlds. In this way, the design is much like the Application/Document/View architecture found in the Microsoft Foundation Classes (a variant of Smalltalk's Model/View/Controller architecture).

The Application class, shown in Figure 2, aggregates the SimulationDate and DatabaseAdministrator classes.

Figure 2   Application Class Diagram - Operations
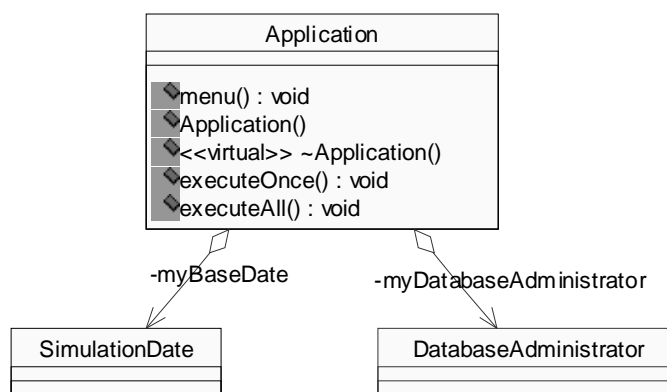
The Application class is responsible for creating the singleton[1] World and DatabaseAdministrator objects. It is also responsible for maintaining the simulation base date and conducting all user interaction.

_____

[1] A singleton is a class that has only one instance.

3.2   The World Class

The `World`, the central class in ILUTE, manages all of the simulated objects in the system. The `World` class, shown in Figure 3, has a simulation date and contains a number of operations related to processing various events and decisions.

Figure 3   World Class Diagram – Attributes
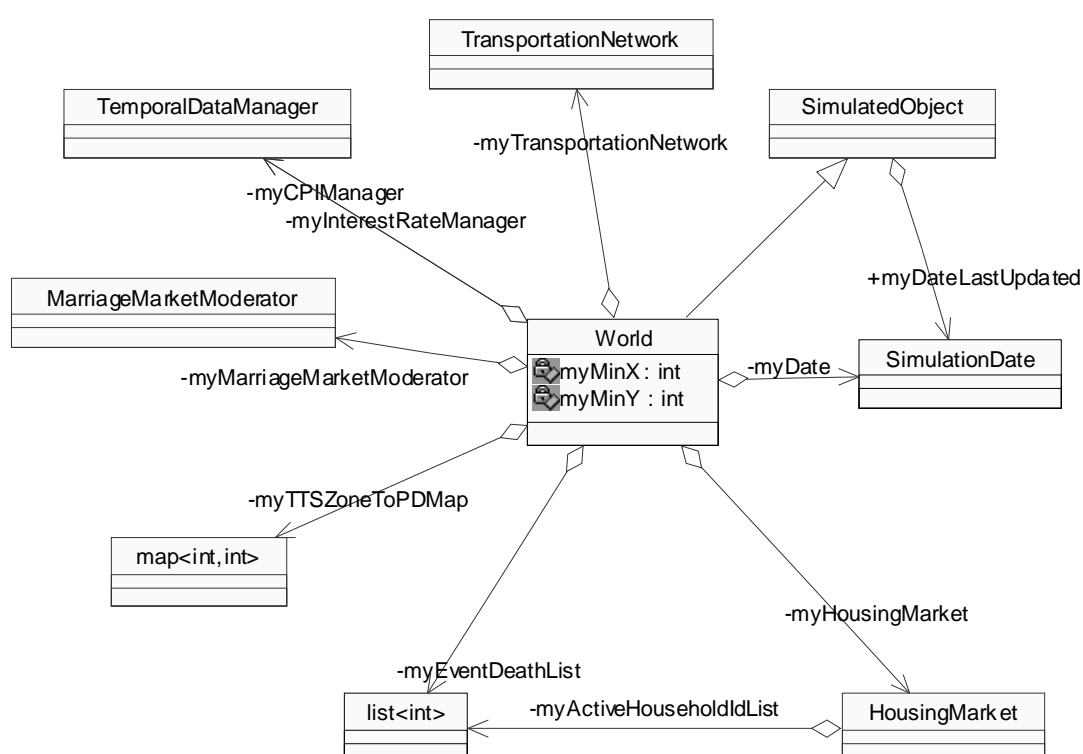

3.3   The Household and Person Classes

The `Household` class is responsible for managing the behaviour of a simulated household. The `Person` class is responsible for managing the behaviour of a simulated person. Figure 4 shows the attributes of the `Household` and `Person` classes and their relationship to several other associated classes.

Early in the development of the model, it was deemed necessary for the model to work either with in-memory data structures or with an external relational database. The class design in the ILUTE prototype is flexible enough to support either approach. Identifiers, called `ids` in the

model, are used to provide a key by which an object can be found either in an in-memory map or in a relational database. This requirement, while adding complexity to the model, offers considerable flexibility to the designer of future submodels. In the current implementation, all data values are stored in memory and the database is only used for making data persistent (for sharing, branching, or analyzing).

Figure 4   Household and Person Class Diagram – Attributes
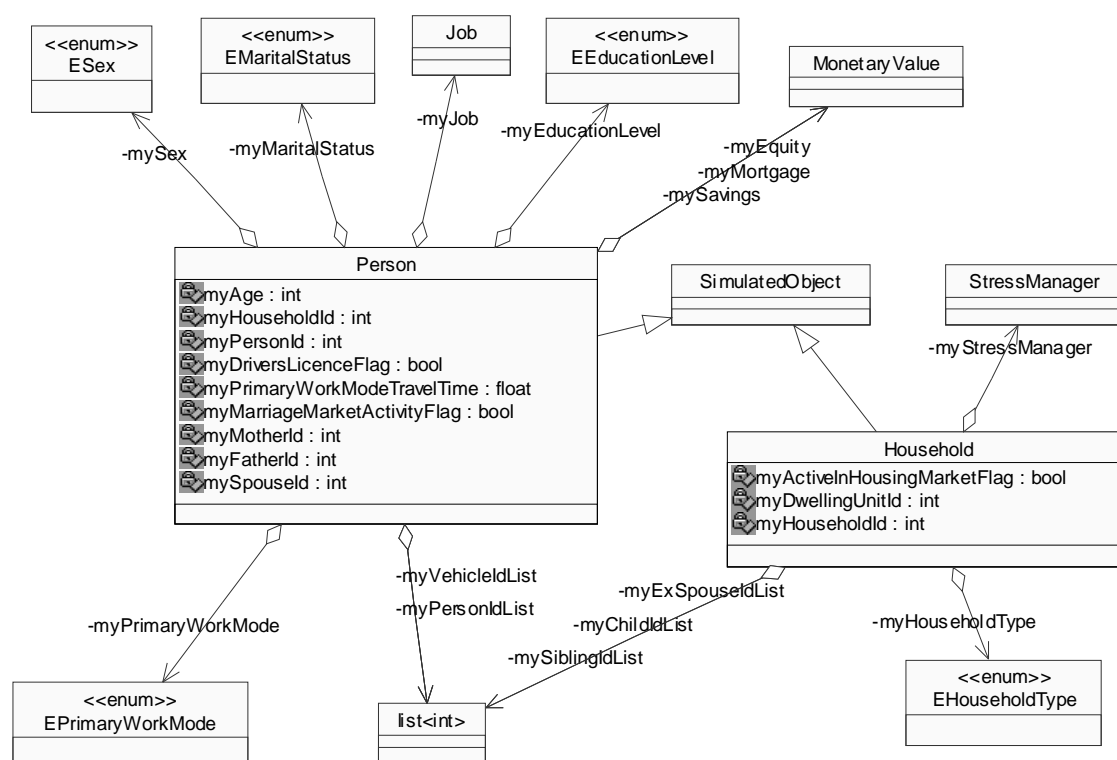
4.   Architecture: Processes

The decision processes in ILUTE provide a flexible and extensible mechanism for handling complex real-world decision processes. This section examines temporal leads and lags, decision-making, and the stressor mechanism.

4.1   Temporal Leads and Lags – Market Disequilibria

During the development of the operational prototype, it became clear that the model needed to support a variety of leads and lags within the system. Moreover, the model needed to avoid the equilibrium assumption/concession made in many other models in order to accurately represent real-world conditions. There are many examples of where leads and lags occur in the system. For example, a house may be on the market for many months before it sells. At any given time, there are a number of houses remaining on the market. As a result, the market is never cleared. Similarly, it takes several years to build an apartment-style condominium building once the decision to develop is made.

Lags and leads in the prototype are handled through time-proxied decision processes that represent future events. The attributes derived from these future intentions are stored in the object's current state and are automatically made persistent when the object's state is stored. A good example of leads and lags occurs in the housing market submodel where the purchase of an unbuilt condo might precede the move-in date by several years. During any time interval, the purchaser may wish to forfeit the deposit and withdraw from the deal. Similarly, the developer may choose to move the closing date back (with possible associated penalties).

As an extension to the mechanism for handling leads and lags, the ILUTE prototype also handles anticipatory behaviour where decisions are made in anticipation of a future change in the state of the system. For example, a family might decide to move to a larger dwelling unit in anticipation of having a baby within the next two years. While it is not possible for an agent to examine the actual future state (doing so would create an infinite recursion), it is possible for an agent to anticipate a future state based on the current system state. For example, a person might have a process to anticipate future interest rates. While this anticipation might affect a number of important decisions, it is purely speculation (even when well-informed) on behalf of the person involved.

While it was initially thought that such anticipatory decisions could just as easily be "picked up" after the fact (i.e. once the baby arrives, they decide that the house is too small and look for a larger house), such an approach results in decisions being made outside of the actual economic context in which they would have occurred. In the case of the move to a larger home, the decision might have been made during a time when interest rates and house prices were low. If those conditions are not the same after the baby arrives, the resulting decision might be very different.

4.2 Accumulators

An accumulator allows a value to grow slowly over time until a given threshold is reached and an event is triggered. The accumulator concept is important in any case where a trigger is not occurring in response to a single change in state, but rather as the cumulative effect of a number of state changes. Writers of individual ILUTE submodels are free to choose whether an accumulator is necessary in capturing such decisions. An example of an accumulator is a bank account (holding of cash assets) where wealth can accumulate in the account over a long period of time. An example of a non-accumulated item is income, which has a particular value at any point in the simulation. The use of the accumulator enables a current measure of a variable that has changed over time. The stressor mechanism presented in the following section provides a good example of the accumulator concept in action.

4.3 Stressors and the Stress Manager

The stressor[2] and stress manager components of ILUTE provide a rich and extensible mechanism for handling triggered events, joint decisions, accumulated stress, and person-household interactions.

The `StressManager` class, shown in Figure 5, is responsible for handling all stress-related processing. While any object can be associated with a stress manager, the present implementation only assigns stress managers to households. When a household is updated, its stress manager is also updated. The update mechanism is responsible for updating all stressors corresponding to that household.

The `StressManager` class manages a vector of `Stressor` objects. Each specific stressor is a subclass of the generalized `Stressor` base class. A sample of the `Stressor` class hierarchy is shown in Figure 6. The ILUTE prototype implements two stressors within its hierarchy: travel time and financial. The implementation of the travel time stressor is relatively complete whereas the financial stressor serves only to demonstrate polymorphism within the design of the architecture. The `Stressor` class is an abstract base class[3] for specific stressors. All stressors have the same interface, as defined in the `Stressor` class.

---

[2] A stressor is something that causes (or can cause) stress.

[3] An abstract class is a class that is never instantiated. It serves to define a common interface for its subclasses.
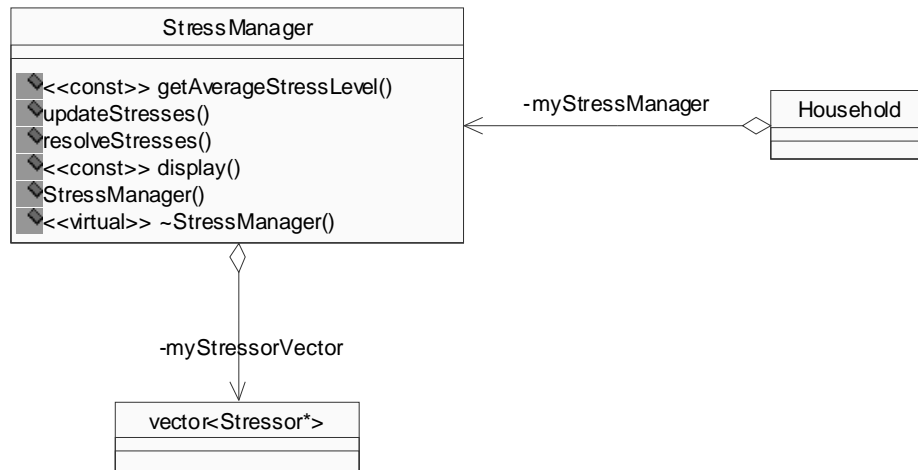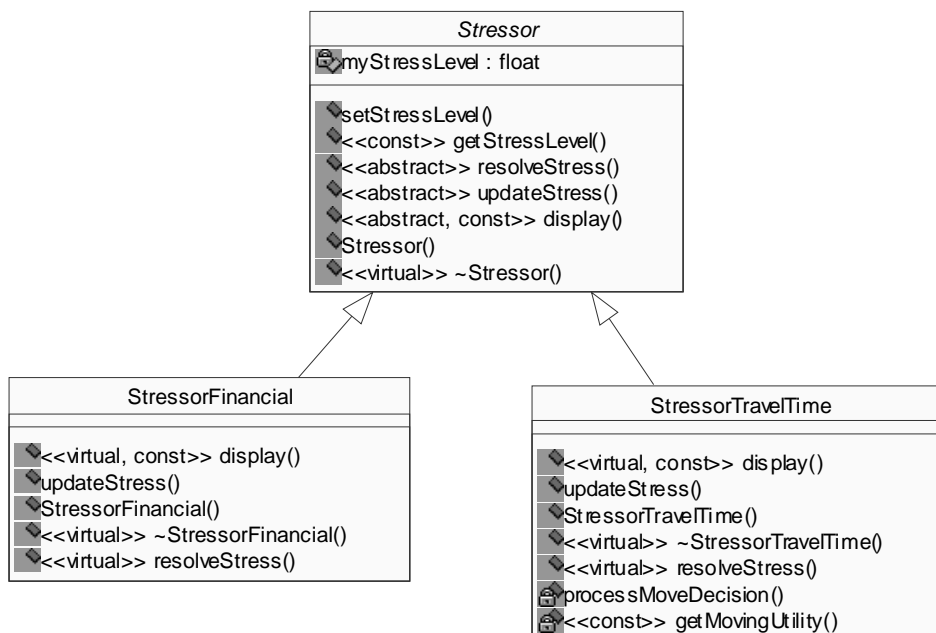
Figure 5   Stress Manager Class Diagram

Figure 6   Stressor Class Diagram

In the current prototype, stress levels[4] are stored as normalized floating-point values between 0 and 1. Stress levels below 0.25 are considered minor; those between 0.25 and 0.75 are considered moderate; and those above 0.75 are considered severe. The stress update mechanism, invoked on all stressors by the `StressManager`, allows the stressors to be updated at any time. As stressors are persistent, the stress level can increase or decrease as the stress accumulates or disperses.

Once a stress is severe, indicated by a stress level above 0.75, a resolution is sought. The automatic triggering of a stressor-specific resolution mechanism could easily be adapted to incorporate a Monte Carlo technique where the likelihood of dealing with a stressor is greater (but not absolute) when it is severe. This approach would seem to better mirror the reality of life where some individuals put up with inordinate amounts of stress (at least for short periods of time).

The stressor-specific resolution mechanism allows each stressor to have its own resolution mechanism. The resolution strategy for travel time, for example, might be to investigate the best option of moving jobs, moving residences, adding a vehicle, or moving and changing jobs. The stress-resolution mechanism can be as simple or as complicated as desired. For example, it would be possible to incorporate risk in each alternative and the stress resolution mechanism could consider the decision-maker's attitude toward risk in evaluating the alternatives.

4.4   Microsimulating Market Interactions

Within ILUTE, a number of processes can be abstracted as market interactions: the purchase of a home or automobile, the selection of a spouse, the decision to select a job, etc. The common theme in all such interactions is the idea that a consumer and a supplier transact within a market. Early attempts to model such interactions involved the simple pooling of suppliers on one side, consumers on the other, and an exchange process that matched consumers with suppliers based on some form of utility maximization. While the initial approach had some conceptual appeal, it tended to over-optimize the matching process.

---

[4] The terms *stress level* and *stressor* are used to avoid the ambiguities in the common use of the term *stress*. In ILUTE, a *stressor* is something that causes *stress* and the *stress level* is a measure of the amount of stress. It may be convenient to think of stress as a disutility.

In the real world, not all consumers and suppliers interact in the same market. For example, it is very unlikely that a home purchaser would examine every house on the market before making a decision. Not only would it be prohibitively time consuming, it would also be virtually impossible because the housing supply is highly dynamic. On any given day, there is a sliding window of opportunity. Similarly, few persons get the opportunity to date all potential partners before making a decision on marriage.

4.5   Activity Scheduling

ILUTE is an activity-based model and one of its key components is the scheduling of activities by persons in the system. Persons engage in activities for a variety of reasons – some personal and some in order to meet household obligations. The activity scheduler in ILUTE's operational prototype is based on a design by Miller and Roorda (2003). Several considerations were observed in the development of this model:

1. Activity scheduling involves sharing of responsibilities and resources – a problem that requires the concepts of communication and collaboration among simulated agents. Competing for scarce resources (such as a single car in a multi-driver household) can also lead to complicated resource use patterns.

2. Many activities require multiple participants. Coordinating the arrival of multiple participants adds considerable complexity to the scheduling dilemma. Moving activities involving multiple participants is much harder than moving individual activities. Some trips, often called *serve dependent trips*, are made to service other persons.

3. While it is convenient to think of activities as isolated events, real-world activities are often connected as projects. Axhausen (1998) defines a *project* as a coordinated set of activities tied together by a common goal or outcome. For example, renovating a basement is a project that involves multiple trip-generating activities.

4. Many aspects of activity scheduling are fuzzy. A shopping trip is flexible in terms of its timing – work is generally less flexible. To catch the last train of the day, however, it is necessary to be on time. Because of the difficulty of predicting travel time to the train station, the person is likely to leave some contingency time and arrive early. While handling such fuzziness is natural to a human, it requires special techniques in the case of a computer model.

5. The generation of activities is not a task that can be completed "a priori" to scheduling as real-life schedules change (often resulting in sub-optimal schedules). The bumping of lower priority events for higher priority ones adds another dimension of complexity. In addition to bumping and possibly rescheduling activities, some activities can be shortened, lengthened, or interrupted. The cost of each rescheduling option, however, can vary considerably with the activity.

6. The utility derived from some tasks is connected to the time that has elapsed since the task was last executed. Much like decreased marginal utility of goods to a consumer, certain activities become more important than others as elapsed time increases. For example, the utility derived from eating, sleeping, or grocery shopping is clearly linked to the elapsed time since the activity was last completed. When some activities are postponed, they rise in priority (utility) until they become critical.

7. The activity scheduling system must handle a wide range of modes including auto driver, auto passenger, commuter rail, transit, express transit, taxi, walk, and bike. Sometimes, mode choice is determined by the schedule and other times the schedule is determined by the mode choice. Regardless of the direction of the effect, the choice of possible modes makes it difficult to create provisional schedules (especially in the case of a shared vehicle resource).

8. In order to assess the sensitivity of certain policy variables, it is important to support ride-sharing modes. Examples include car and van pooling, as well as auto passenger commutes where a passenger is dropped off "on the way" to work. An even more difficult case to handle is the emergence of "slugging" in Washington D.C. and other congested urban areas. Riders (slugs) wait in "slug lines" at an express bus stop (with a slug sign) and are picked up by a driver going in the same general direction. This semi-formal mode of ride sharing has yet to be modelled.

9. Some activities are more likely to require a vehicle than others. Shopping, for example, is often a task that is conducted with a personal auto. Shopping by other modes is likely to affect the frequency and outcome of the shopping trip. A trip to the grocery store by taxi, for example, might cause one to buy more groceries but shop less frequently. Similarly, shopping trip by public transportation, walking or biking might result in more frequent trips with smaller purchases on each outing.

10. Some activities have a "natural frequency". For most regular workers, working is a daily weekday requirement whereas grocery shopping is perhaps a weekly event.

5.   Architecture: Data

5.1   Data Synthesis

The data synthesis procedure in the ILUTE prototype is quite extensive, but is not designed to provide a census-calibrated population. Rather, the intent of the prototype procedure is to generate a comprehensive population of a prescribed size for the purpose of testing the prototype. As a parallel research effort for ILUTE, several other researchers are developing a formal prototype synthesis procedure for generating a representative population from census data.

There are a significant number of interconnections within the system that must be maintained during synthesis. The synthesis procedure is an optional user-initiated procedure as the model can also be run on existing data. When the user chooses the synthesize data option, the `World` is asked to synthesize a given number of households. The number of families, persons, dwelling units, jobs, vehicles, and buildings synthesized is a function of this initial number.

The data synthesis procedure in the prototype consists of four sequential phases: the synthesis of households and persons, the synthesis of buildings and dwelling units, the assignment of households to dwelling units, and the assignment of the primary work mode.

5.2   External Temporal Data

The `TemporalDataManager` class, shown in Figure 7, imports and manages time-varying data tables stored in delimited text files. Historical consumer price index data and historical cost of borrowing data are stored in two such files. A number of submodels and processes use external temporal data either directly (e.g. residential mobility submodel) or indirectly (e.g. auto transaction submodel). As it was anticipated that some sources of data would be required in multiple submodels, a formal mechanism for handling external temporal data was developed.

The temporal data manager in the ILUTE prototype is both efficient and flexible. Efficiency is achieved by storing only the original data values found in the external data files. Flexibility is achieved by providing a number of functions for retrieving and interpolating the data from the in-memory data store. The separation of interface and implementation allows different submodels to interact with the same data in different ways. The temporal data values are

stored in a C++ Standard Library *map* data structure. This structure enables the efficient look-up of a value corresponding to a given key.

Figure 7   Temporal Data Manager Class Diagram
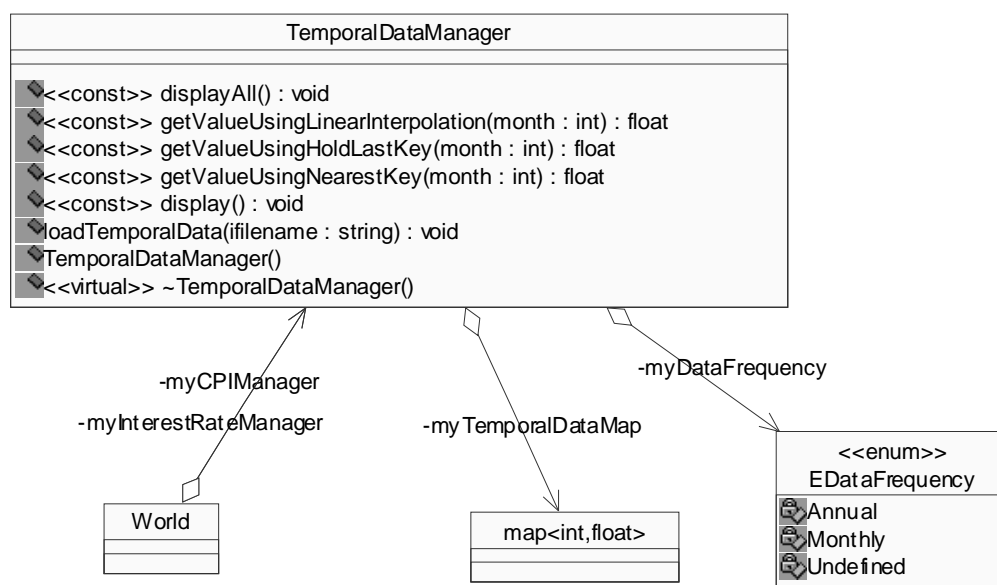
In urban systems models, external data files often contain a combination of historical and forecast data[5]. Forecast data are generally not available with the same frequency as historical data. As a result, there are often gaps in the available data. For example, historical consumer price index data might be readily available on a monthly basis but forecast data might only be available on an annual basis. One of the jobs of the temporal data manager is to fill these gaps using one of several interpolation techniques.

5.3   Monetary Values

The changing value of money over time presents an interesting challenge for the model. In a long-run simulation (e.g. 30 years), changes in the value of money can significantly distort the outcome of the model if not handled carefully.

_____

[5] It does not matter if these historical and forecast data are stored in one file or two files.

In the ILUTE operational prototype, all monetary amounts are stored as a pair of numbers consisting of a date-value pair. For example, the pair ($75.00, 1985) would correspond to 75 dollars in 1985, which might have the same spending power as the pair ($100.00, 1992). The monetary value structure automatically encapsulates both data members. When a monetary amount is used, it can be converted to current-day dollars as required. Comparison and subtraction operators can automatically convert monetary values to common units before performing calculations. This approach blends the benefits of the above two approaches as it does not require regular updating and dollar amounts can be kept in the unit that is most reasonable. Figure 8 provides an overview of the `MonetaryValue` class and its conversion operations.

The auto transaction model in the operational prototype exemplifies the use of the monetary value classes as it assumes that all dollar amounts are in 1998 Canadian dollars. The coefficients of the auto transaction equations would need to be changed if non-1998 dollars were used.

Figure 8   Monetary Value Class Diagram
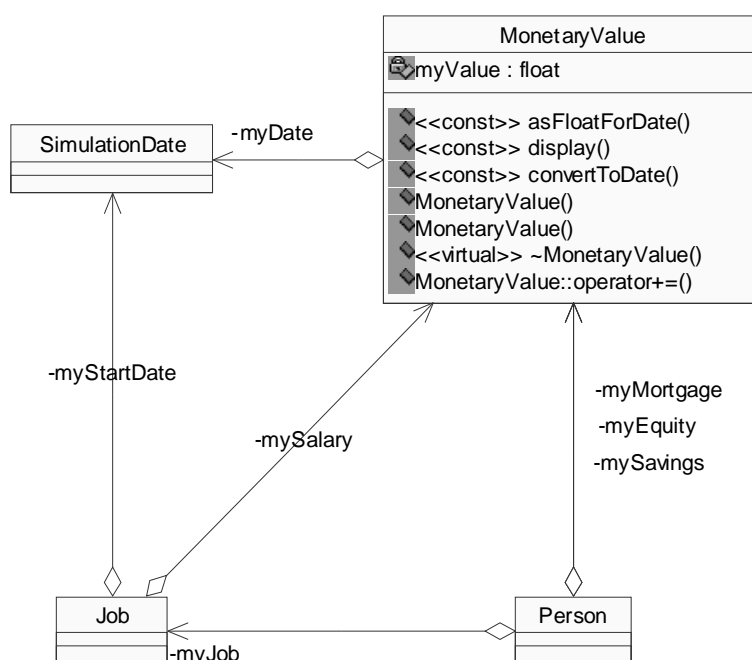
5.4   The Database Administrator Class

The `DatabaseAdministrator` class, shown in Figure 9, is responsible for the persistent storage and retrieval of simulation data. The database administrator interface allows the ILUTE prototype to create and drop database tables, load the system state, and save the system state. A table suffix is used to specify the month for which the data are valid. The database administrator communicates with a `CDatabase` object from the ODBC Database Support library to carry out low-level database session commands. The `DatabaseAdministrator` is a singleton class with the only instance being managed by the `Application` class.

Figure 9   DatabaseAdministrator Class Diagram
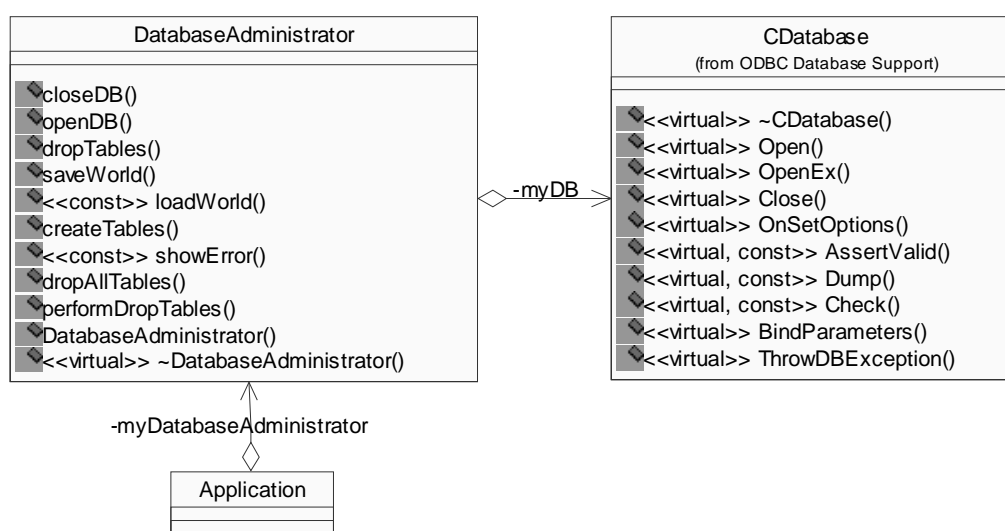
5.5   The Record Set Subclasses

A number of subclasses of the `CRecordset` class are used to perform the actual transfer of state information to and from the database. These classes override the virtual interface shown in Figure 10. For simplicity, this diagram only shows the interface of the base class. The derived classes, one per ILUTE agent, implement the actual methods for data exchange between the in-memory representation and the on-disk representation. The use of parallel `CRecordset` subclasses makes it very easy to manage and extend the database classes.

Figure 10   The CRecordset Class Diagram
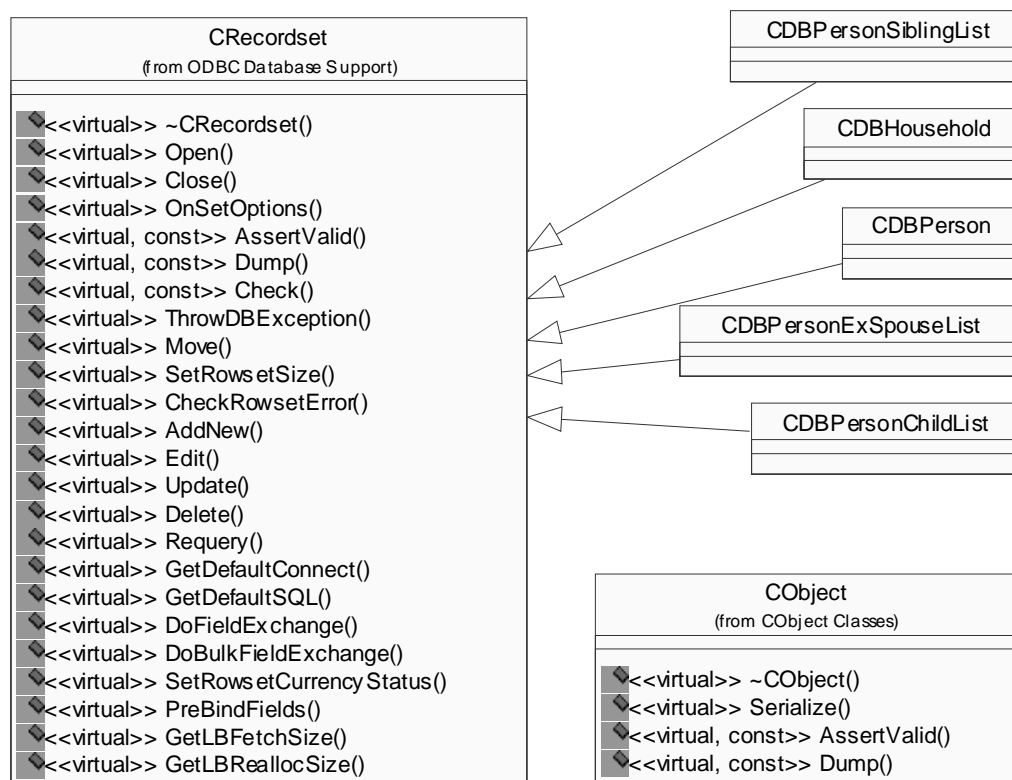
6.   Submodels

As a formal test of the integrity and extensibility of the ILUTE prototype, three submodels of representative complexity were implemented: the Auto Transaction submodel, the (Resale) Housing Market submodel, and the Activity Generation submodel.

6.1   Housing Market Submodel

The Housing market submodel is responsible for processing all of the households that are active in the housing market. These households include both those looking to rent as well as those looking to purchase. ILUTE uses a three-step process for modelling residential mobility: 1) a mobility decision, 2) a search process, and 3) a bid. Mobility decisions are generally triggered by a stress manager, but may also be triggered on a random basis. Once the decision

to enter the housing market has been made, it is the role of the housing market submodel to complete the search and bidding processes. It should be noted that the decision to become active in the housing market does not force a household to follow through with a purchase.

In the ILUTE architecture, the `World` class manages the housing market (it contains the singleton `myHousingMarket` object, which is the sole instance of the `HousingMarket` class). The `HousingMarket` class maintains a list of households that are active in the housing market. The `HousingMarket` class also contains a number of processes that are responsible for managing the housing market. The list of dwelling units available for sale is stored inside the `Neighbourhood` class.

6.2   Auto Transactions Submodel

The theoretical (mathematical) portion of the auto transactions submodel was developed by Kouros Mohammadian (2002). The auto transactions submodel is a properly estimated empirical model that uses a nested logit equation with calibrated real-world parameters. The submodel uses attributes of the households, owners, drivers, their current vehicle bundle, and the attributes of each vehicle to determine if the household will maintain its existing auto ownership level, purchase a vehicle, dispose of a vehicle, or trade a vehicle.

The auto transactions submodel was converted from its theoretical form to C++ code as a test case for submodel implementation. As a representative model, it is anticipated that many models can be implemented in the same fashion. As with all fully evolved submodels, the auto transactions submodel is very demanding in terms of its data requirements. Moreover, it is also representative of models that have been calibrated with data from a fixed year.

6.3   Activity Submodel

The activity generation submodel was added as a test case for household and person activity scheduling. Specifically, it was added to ensure that the widely varying temporal needs of the main model and the activity submodel could be properly accommodated. In the current ILUTE operational prototype, the `Activity` class is at the core of the activity generation submodel. Activities occur in space and time and have various scheduling dependencies. A variety of activity types were incorporated into the submodel including work, school, shopping, other, and free time. The integration of a more formal model for activity participation is a likely next step in the evolution of the ILUTE prototype.

6.4   Output and Visualization

One of the most challenging problems in working with complex microsimulation systems is understanding what is happening in the system. Microsimulation models provide an incredibly powerful ability to follow the lives of individual entities as they move through the system. Understanding the cumulative effect of these individual lives, however, is a different matter entirely. While aggregate statistical measures are traditionally used to understand the changes that occur in a complex system over time, newer technology for visualizing these changes using computer animation are emerging.

Figure 11   Sample Rendering of Night Travel Time to the CBD
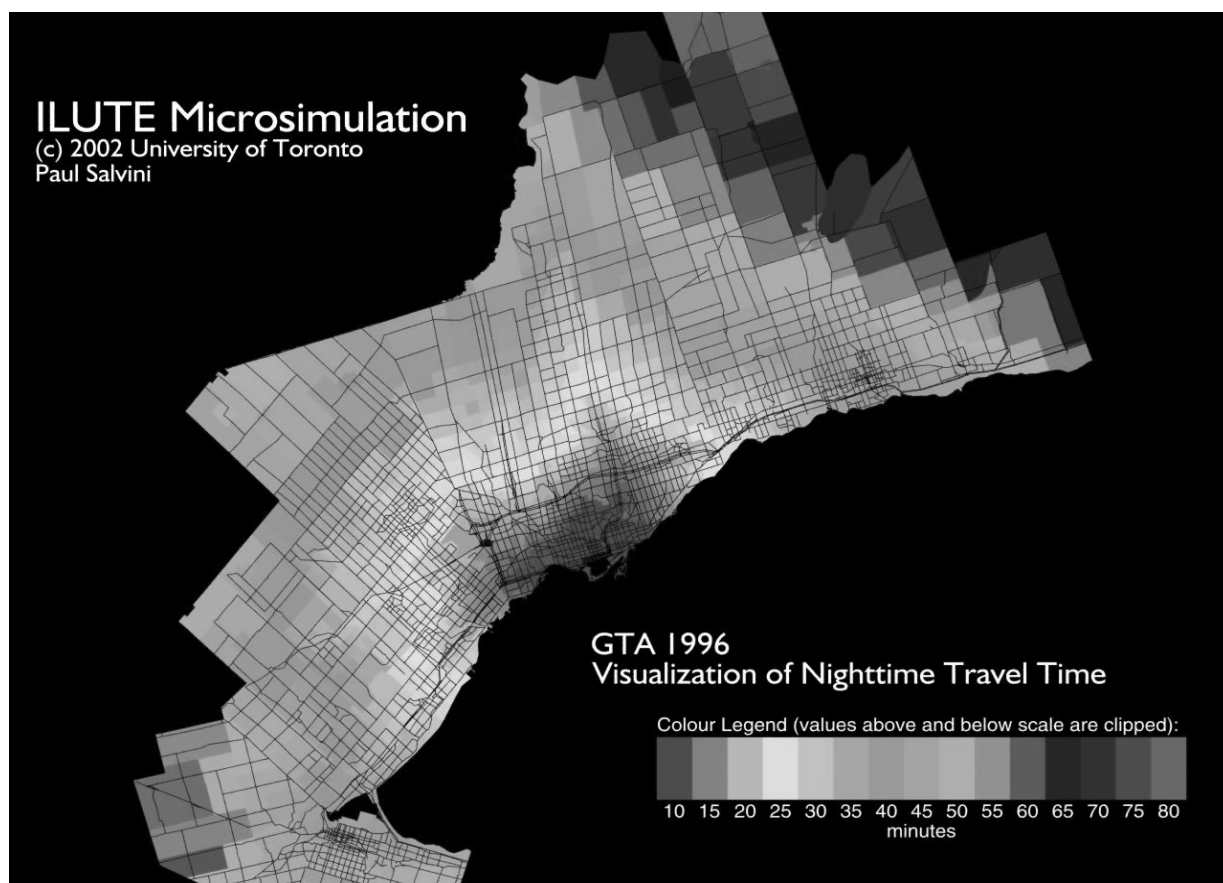
ILUTE seizes the opportunity to explore this potential advance by outputting to a 3D animation software package called Houdini[6]. In the ILUTE prototype, an animated 3D surface is used to visualize the behaviour of the model with additional information conveyed by colour and overlaid features. Key frames are established at fixed times (e.g. one year intervals) and the software automatically generates intermediate frames to animate the changes from one time step to another. The amount of information that can be compressed into a single 3D animation is impressive. The model uses several techniques to maximize the amount of information delivered to the user. Figure 11 shows a static, false colour rendering of travel times (to the CBD) superimposed on a map of the GTA. In the animated version, the colours shift as the travel times change over time. The year also updates to show the current year represented by the animation. As early tests with changing legend scales proved less intuitive to the viewer, the current implementation uses a fixed legend scale.

7. Future Work and Concluding Remarks

While the prototype described in this paper has created a solid foundation for future ILUTE researchers, there is a vast amount of work that remains to be done. This section proposes a number of future research and development tasks that will help take ILUTE from an operational prototype to a fully functioning urban systems model:

1. The TASHA activity scheduler (Miller and Roorda, 2003) should be integrated with the ILUTE prototype to replace the current activity generator.

2. Census-calibrated population synthesis routines (Guan, 2001, 2002) should be integrated with the ILUTE prototype to supplement the current synthesis routines. It is recommended that the current routines be left in place as they enable the development of small test sets. Once integrated, the synthesis routines need to be expanded to generate object types other than persons and households (e.g. buildings, dwelling units, jobs, firms, establishments, etc.).

3. The housing supply side, based on Haider (2003), needs to be implemented to ensure that housing supply keeps up with housing demand. In the current ILUTE prototype, housing becomes scarce as the population grows and the housing supply does not. Similarly, other supply side models need to be incorporated.

_____

[6] A non-commercial version of this software is available for free download – see www.sidefx.com for more information.

4. The ILUTE model currently represents jobs, but has no representation of firms and establishments. Submodels for these classes are under development by other members of the ILUTE team (Douglas Hunt and John Abraham at the University of Calgary). The integration of these submodels will provide the analogue to the person and household classes currently implemented in the ILUTE prototype.

5. It would be beneficial to calculate travel times directly within the model (or with a dynamically integrated third party package). Changes within the urban environment (including changes to the transportation network itself) should be reflected in travel times[7] as the use of out-of-date travel times can hurt the model's accuracy. Manually running a new network assignment after every time step is prohibitively expensive. More generically, support for the "live" exchange of in-memory data (through third-party APIs, for example) would enable the efficient integration of third-party packages.

6. In the ILUTE prototype, a mechanism exists to adjust prices from one time period to another. However, this mechanism does not account for relative differences in the price of different types of goods over time. For example, gasoline prices do not necessarily track the overall consumer price index. Having separate consumer price indexes for different types of goods would solve this problem.

7. Future work for the temporal data manager includes implementing a spline interpolation option and an extrapolation option.

8. While it should be a matter of straightforward development (the framework is largely in place), the ILUTE prototype needs to be extended to handle temporally varying spatial areas.

9. The current ILUTE user interface was written for test purposes only. The formal ILUTE interface should allow the user to manage simulation runs and model parameters. Data logging and agent tracing mechanisms also need to be exposed within the interface.

10. The ability to visualize spatiotemporal data as an animated 3D surface opens the door to wonderful possibilities. Finding the most visually meaningful way to show each

---

[7] Although lagged travel times are useful in activity scheduling to represent a person's travel time experience.

type of research phenomenon will be a challenging but exciting task for future research.

11. Once the model has progressed to the point where it contains real-world data, the job of calibration and validation testing will begin. While ILUTE attempts to endogenize as much behaviour as possible, some submodels remain externally calibrated. Testing the stability of the ILUTE model is also an important consideration as multiple model runs (in cases where random effects exist) might result in significantly different end results. While earlier research suggests that the main model will be relatively stable (Salvini, 1998), the individual submodels may serve to destabilize the end product.

7.1   Concluding Remarks

The development of a comprehensive dynamic microsimulation model of urban systems is by no means a small task. The success or failure of the overall model will depend largely on the ability of the individual submodels to credibly capture the behaviour of real-world entities. It will also depend on the ability of these submodels to communicate through the main ILUTE framework. While the operational prototype presented in this thesis represents an important early step in the development of a working model, much work remains.

It is uncertain how well this model will capture the actual behaviour of our urban areas. Indeed, it is even unclear if this new generation of models will do much better than the early models at long-run forecasting. While the model is definitely a more accurate representation of our real world, it is unknown whether this new level of accuracy is sufficient. But lest the end product be discouraging, it is important to note that the journey itself holds great promise. Developing ILUTE presents a rare opportunity to look inside a very complex system and understand, at least at a basic level, its intricate and interconnected nature.

For transportation engineers, ILUTE provides the opportunity to explore transportation's role in shaping an urban system. Specifically, it allows transportation issues to be examined within a broader context. For planners, ILUTE provides an opportunity to experiment with many of the variables that affect urban systems and understand the relationships between them. For others, and perhaps even for the public, it is a chance to understand that the consequences of a given change are not always easy to predict. Even if the results do not ultimately match those of the real world, a wealth of insight is sure to be obtained.

As a final thought, perhaps it is worth taking a few steps back to contemplate the implications of having a perfect urban systems model. If we could accurately predict the future of an urban

area, how would we use that information? Would we be able to agree on a common vision for what constitutes an ideal urban system? Would our society be willing to make individual sacrifices on the promise of a better future? Will we be able to balance our short-term desires with our long-term needs?

References

Axhausen, K. (1998), Can We Ever Obtain the Data We Would Like to Have?, in T. Gårling, T. Laitila and K. Westin (eds.) _Theoretical Foundations of Travel Choice Modelling_, Oxford: Pergamon Press.

Guan, J. (2001). Population Synthesis for the ILUTE Model, working paper, Toronto: Joint Program in Transportation, University of Toronto.

Guan, J. (2002). Synthesizing Family Relationships Between Individuals for the ILUTE Micro-simulation Model, B.A.Sc. thesis, Toronto, Department of Civil Engineering, University of Toronto

Haider, M. (2003), Integrated Land Use Transportation Modelling of Housing Starts, Ph.D. Thesis, Graduate Department of Civil Engineering, University of Toronto, Toronto.

Miller, E. J., D. S. Kriger and J. D. Hunt (1998), Integrated Urban Models for Simulation of Transit and Land-Use Policies, Final Report, Transit Cooperative Research Project H-12.

Miller, E.J. and M. J. Roorda (2003), A Prototype Model of Household Activity/Travel Scheduling, forthcoming in Transportation Research Records, the Journal of the Transportation Research Board.

Miller, E. J. and P. A. Salvini (2001), The Integrated Land Use, Transportation, Environment (ILUTE) Microsimulation Modelling System: Description & Current Status", Chapter 41 in D. Hensher (ed.) The Leading Edge in Travel Behaviour Research, selected papers from the 9th International Association for Travel Behaviour Research Conference, Gold Coast, Queensland, Australia.

Mohammadian, A. (2002), Dynamic Modeling of Household Automobile Transactions within a Microsimulation Framework, Ph.D. Thesis, Graduate Department of Civil Engineering, University of Toronto, Toronto.

Salvini, Paul (1998), The Architectural Design Of ILUTE, An Integrated Dynamic Microsimulation Modelling Framework, M.A.Sc. Thesis, Graduate Department of Civil Engineering, University of Toronto, Toronto.

Salvini, Paul (2003), Design and Development of the ILUTE Operational Prototype: a Comprehensive Microsimulation Model of Urban Systems, Ph.D. Thesis, Graduate Department of Civil Engineering, University of Toronto, Toronto.