SmartDb: An Object-Oriented Simulation Framework for Intelligent Vehicles and Highway Systems *

Aleks Göllü, Akash Deshpande, Praveen Hingorani, and Pravin Varaiya
Department of Electrical Engineering and Computer Sciences
University of California, Berkeley, CA 94720

Abstract

SmartDB is a framework for the uniform specification, simulation, optimization, evaluation, and implementation of Intelligent Vehicle Highway System (IVHS) alternatives. The salient concepts in SmartDB are: 1) layered control architecture, 2) coordination of distributed control agents through communication, 3) combined discrete and continuous dynamical systems, known as hybrid systems, and their control and verification, 4) object oriented simulation, and 5) distributed and open architecture. This paper summarizes the first three concepts and describes in detail the simulation constructs and the distributed and open architecture of SmartDB.

1 Introduction

Highway congestion is imposing an intolerable burden on many urban residents. It is estimated that lost productivity due to traffic congestion costs $100 billion each year in the United States. Alongside congestion, safety continues to be a prime concern. In 1991, 41,000 persons died in traffic accidents, and more than 5 million persons were injured.

*Research supported in part by the National Science Foundation and the California PATH program in cooperation with the State of California and US DOT. The contents of this report reflects the views of the authors who are responsible for the facts and accuracy of the data presented herein

Intelligent Vehicles and Highway Systems (IVHS) is a comprehensive program initiated by the U.S. Government under the Intermodal Surface Transportation Efficiency Act of 1991 to improve safety, reduce congestion, enhance mobility, minimize environmental impact, save energy, and promote economic productivity in the transportation system. The IVHS program combines several modern technologies, including information processing, communications, control, and electronics. IVHS has the following sub-programs.

Advanced Traffic Management Systems
: ATMS provides subsystem integration of traffic management and control systems, and performs real-time traffic control to respond to dynamic traffic conditions.

Advanced Traveler Information Systems
: ATIS acquires and analyzes information about transportation network dynamically, and communicates advisory information to travelers.

Advanced Vehicle Control Systems
: AVCS uses computers, communications, and control systems in the vehicles and the highways to enhance vehicle control.

Commercial Vehicle Operations
: CVO improves the safety and efficiency of commercial vehicle and fleet operations.

Advanced Public Transportation Systems
: APTS integrate public transportation with vehicle-highway systems by using component technologies from other functional areas.

Apart from the U.S., there is substantial IVHS activity in Europe under the PROMETHEUS[1] and the DRIVE[2] projects, and in Japan under the RACS[3], AMTICS[4] and VICS[5] projects.

[1] Program for European Traffic with Highest Efficiency and Unprecedented Safety
[2] Dedicated Road Infrastructure for Vehicle Safety in Europe
[3] Road/Automobile Communication System
[4] Advanced Mobile Traffic Information and Communication System
[5] Vehicle Information and Communication System

Modeling and simulation have been identified as important steps in realizing these transportation initiatives. The IVHS strategic plan [4] requires modeling and simulation in the following areas: urban traffic network models, traffic system models, vehicle-road models, driver-vehicle models, traffic models with dynamic traffic assignment, driving scenario simulation, and advanced vehicle control systems (AVCS) architecture simulation. A framework in which IVHS alternatives can be specified, simulated, and evaluated uniformly is crucial for objective comparison of the proposed alternatives. Such a framework must also aid in the implementation of the selected alternative.

SmartDB is a framework for uniform specification, simulation, optimization, evaluation, and implementation of IVHS alternatives. *SmartDB* is targeted at the Advanced Vehicle Control Systems (AVCS) functions. At the same time, it also addresses the automation requirements of the other functional areas. In particular, *SmartDB* meets the modeling and simulation needs of urban traffic network models, traffic system models, vehicle-road models, traffic models with dynamic traffic assignment, and AVCS architecture simulation.

The concepts for *SmartDB* have emerged from the SmartPath project at the California PATH Laboratory at the University of California, Berkeley [3]. *SmartDB* is an object-oriented distributed processing simulation environment that can be scaled to meet the performance requirements of large-scale applications. *SmartDB* is an open system that can interface with other simulation environments. *SmartDB* decomposes the IVHS modeling and simulation problem into the following stages:

1. parametrized modeling of the physical system and the control agents in an object oriented semantic data model,
2. simulation of the discrete and continuous behavior of all the objects in the model, and optimization of the model parameters,
3. evaluation of system performance according to specified criteria, and
4. model validation and implementation of selected control strategies for deployment.

These stages are shown in Figure 1.

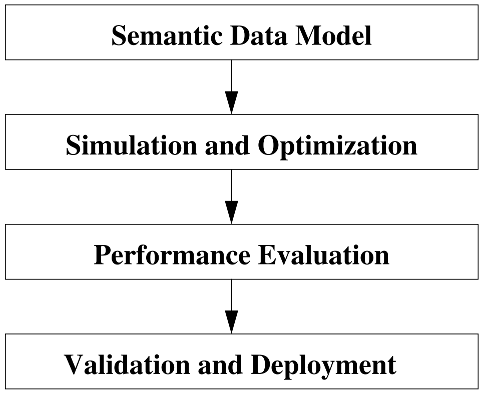

Figure 1: *SmartDB* Stages.

1.1 Semantic Data Model

The *SmartDB* semantic data model captures the relevant aspects of the physical world in a logical model. The logical model consists of software objects that represent physical components. For example, it models vehicles, highway segments, engines, brakes, sensors, and other physical system components as objects. The objects in the logical model have semantic content corresponding to their characteristics, interrelationships, constraints, and behaviors.

1.2 Simulation and Optimization

A simulation run occurs when all objects in a given highway and traffic configuration execute their state evolution behavior. Since object characteristics are parametrized, the model parameters can be tuned during a simulation run based on specified optimization criteria. Such a tuning leads to an on-line optimization of the control strategy.

1.3 Evaluation

The state evolution of the objects during a simulation run can be observed using specialized monitor objects. The monitor objects generate performance results based on specified evaluation criteria. These results are used for an objective evaluation and comparison of alternative control strategies for IVHS. Once a control architecture is selected, *SmartDB* can be used for its rigorous simulation, optimization, and implementation.

1.4 Validation and Deployment

SmartDB's object-oriented approach simplifies model validation and modular deployment. Models for physical objects such as sensors and communication equipment can be validated by replacing software objects in the logical model by the corresponding physical components. *SmartDB* behavior in the two cases—using the software objects and using the physical components—can then be compared to validate the logical model and the controller design.

Similarly, *SmartDB*'s object-oriented approach simplifies modular deployment of the selected control architecture. Once the selected control architecture is simulated and optimized, each software control object that models the behavior of a specific control agent in the physical world can be implemented independently on physical components for deployment.

2 Concepts and Functions

2.1 *SmartDB* Concepts

The salient concepts in *SmartDB* are given below:

1. layered control architecture,
2. coordination of distributed control agents,
3. combined discrete and continuous dynamical systems, known as hybrid [8] systems, and their control and verification,
4. object-oriented simulation, and
5. distributed and open architecture.

2.1.1 Layered Control Architecture

In the layered control architecture proposed by Varaiya and Shladover [7, 6], vehicles perform simple maneuvers such as merging into platoons, splitting from platoons, following the leader, changing lanes, and entry and exit. A vehicle accomplishes complex end-to-end trajectories by performing a sequence of such simple maneuvers. Efficient transportation throughput is achieved by tuning traffic parameters such as platoon size and vehicle speed. The control strategies for such behavior are organized into the following layers: regulation layer, coordination layer, link layer, and network layer. These layers are shown in Figure 2.

Given a maneuver to perform, the vehicle follows a control strategy that regulates its dynamical behavior to a trajectory permitted by that maneuver. Such control strategies constitute the regulation layer. The maneuver to be followed by a vehicle at a given time is determined by coordinating with other vehicles in the neighborhood. The control strategies used for such coordination constitute the coordination layer. The control strategies adapt their behavior based on information about highway traffic conditions. The traffic conditions on highway segments are monitored and controlled by road-side control elements. These are collectively known as the link layer. Finally, information from individual highway segments is aggregated, and end-to-end routing and congestion control is accomplished in the network layer.

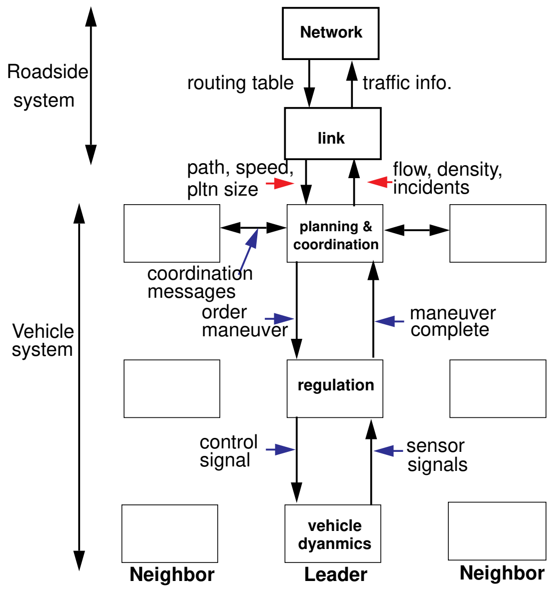

Figure 2: Layered Control Architecture.

SmartDB allows the specification of this and other layered control architectures.

2.1.2 Coordination of Distributed Control Agents

The layered scheme described above yields a distributed control strategy since each vehicle and each highway segment is responsible for its own control. At the same time, effective coordination of these distributed control agents is essential for efficiency and safety. The agents coordinate by following simple heuristic rules. For example, when a vehicle senses another vehicle ahead of it, it requests a merge with it to form a platoon. Forming a platoon increases the efficiency of the highway. However, if the leading vehicle is already a part of a large platoon, it may refuse the merge request since inordinately large platoons are potentially unsafe. Such coordination strategies are modeled using a Discrete Event System (DES) approach. The control agents communicate the discrete events to each other based on coordination protocols. Thus, communication mechanisms are essential both for gathering sensory information and for executing these coordination protocols.

SmartDB allows the specification of sensors, transmitters and receivers, and of communication protocols.

2.1.3 Verification and Control of Hybrid Systems

Whereas the coordination strategies deal with discrete events, regulation strategies deal with continuous evolution. For example, if a merge maneuver is to be executed, then the regulation layer controller must first accelerate the vehicle, close the distance between itself and the vehicle ahead of it, and finally decelerate and follow at the same speed while maintaining a safe distance in between. It is clear that acceleration and braking, speed and distance are continuous parameters that evolve in continuous time. Thus, the discrete coordination event corresponding to the merge command, and the continuous regulation law corresponding to the merge trajectory must be dealt with together. A hybrid system approach is used to model the combined discrete and continuous behavior.

SmartDB allows the specification of both discrete and continuous behavior.

2.1.4 Object-Oriented Simulation

SmartDB is an object-oriented [1, 2] software framework [5]. The object oriented approach is used to construct a logical model of the physical components and their control agents. The objects in the logical model have semantic content corresponding to their characteristics, inter-relationships, constraints, and behaviors. The object-oriented approach simplifies model validation and system implementation for deployment. The object-oriented model is described in section 3.1.

2.1.5 Distributed and Open Architecture

SmartDB is a distributed processing simulation environment that can be scaled to meet the performance requirements of large-scale applications. *SmartDB* is an open system that can interface with other simulation environments. The system architecture is described in section 3.3.

2.2 *SmartDB* Functional Categories

SmartDB is designed to perform the following functions:

Configuration Management—
: the ability to specify a highway network configuration, the traffic patterns on it, and the vehicle and traffic control strategies;

Fault Management—
: the ability to detect faults and significant events such as accidents and congestion, and to respond to them with graceful degradation of highway performance and with automatic fault recovery;

Performance Management—
: the ability to track, optimize, and fine-tune the transportation system performance;

Planning Management—
: the ability to specify and simulate alternative highway and traffic configurations and control strategies for the purpose of planning;

Resource Management—
: the ability to provide an inventory of all highway and vehicle resources and to schedule them for preventive maintenance;

Accounting Management—
: the ability to specify tolls and taxes, and to account for highway usage;

System Management—
: the ability to manage the resources of the *SmartDB* system for multi-user and multiprocessor operation.

3 *SmartDB* Implementation

3.1 Object Architecture

All *SmartDB* objects consist of state, state evolution, interface, input, and output components. This is shown in Figure 3. An object's external interface is defined by its input and output specifications. So long as this interface is met, the state and state evolution of an object can be reimplemented.

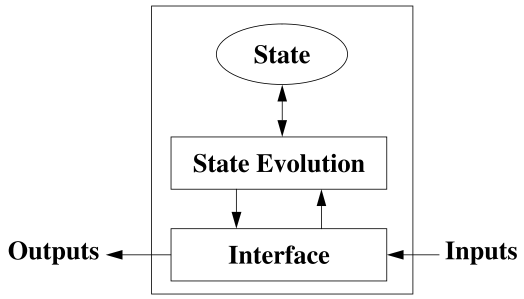

Figure 3: *SmartDB* Object

SmartDB objects are arranged according to a set of connection and containment rules to create aggregate objects. The aggregate objects are also *SmartDB* objects except that their state and state evolution components are implemented by other objects.

It is envisioned that highway automation will be achieved by adding sensors, transmitters, receivers, and control modules to the highway and the vehicles.

To this end *SmartDB* provides a particular aggregate object called "Smart Object" described in Figure 4.

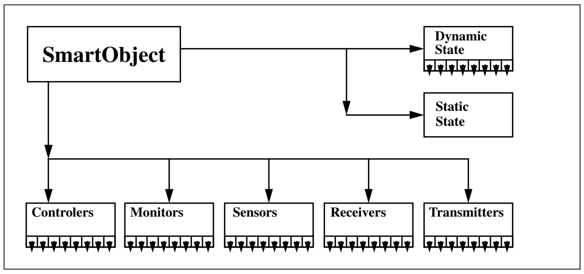

Figure 4: Uniform Representation of IVHS Objects.

Smart objects enable the user to create control and communication hierarchies in a structured manner and consist of the following components:

- static state such as lane width, and vehicle weight, that does not change during a simulation run
- dynamic state such as vehicle speed and lane density,
- state evolution behavior,
- control objects such as regulation objects that determine speed,
- monitor objects for observing state evolution, e.g., a gas tank agent that monitors the amount of carbon-monoxide produced,
- sensors for providing information about the environment, e.g., distance to vehicle in front,
- transmitters and receivers for communicating with neighboring objects,
- inputs and outputs.

3.2 *SmartDB* Functions

The current version of the *SmartDB* implementation supports the configuration, fault, and performance management functions of IVHS. We now describe how the object data model implements these categories.

3.2.1 Configuration Management

SmartDB provides a set of highway objects that can be interrelated according to a set of connection and containment rules. Any possible highway configuration can be created using these objects and their interrelationships. The transportation system is divided into zones. Each zone contains multiple highway segments interconnected using junctions. The highway segments are terminated using traffic sources and sinks. The highway segments consist of sections, entries, and exits. Junctions and sections are divided into lanes. These building blocks and the graphical editor used to create highway configurations are shown in Figure 5.

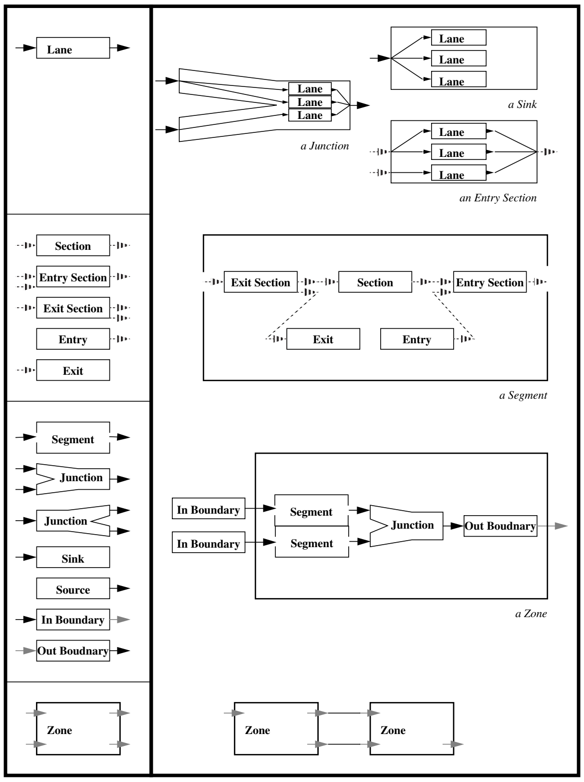

Figure 5: *SmartDB* Graphical Editor for Highway Building Blocks

Incoming traffic to the automated highway is generated by traffic generators in the source and entry objects. Traffic leaving the automated highway is absorbed by traffic absorbers in the exits and sinks. Traffic generator and absorber objects are parametrized to create different traffic patterns. When generators and absorbers are replaced by gateway objects *SmartDB* can interface with packages such as NetSim and Integration.

Vehicle and traffic control strategies are specified by configuring the control, communication, and sensor components of the relevant smart objects.

3.2.2 Fault Management

The simulation framework provides fault detection as well as fault creation mechanisms.

If an object fails to remain within system constraints corresponding fault events are created. System constraints correspond to states the system should not enter. For example a vehicle has to stay within highway boundaries, similarly two vehicles can not occupy the same space at the same time. *SmartDB* creates "accident" events when these constraints are violated. Other constraints such as maximum acceleration or lane change direction in a section can also be specified by the user. Monitor objects are used to detect the violation of constraints. For example a lane change monitor can be used to raise a fault if a vehicle performs an illegal lane change.

Faults such as communication failures or accidents can be created to ensure that the control objects can respond to them with graceful performance degradation and automatic fault recovery.

3.2.3 Performance Management

Monitor objects are used to collect statistical information about the system and to create performance reports. Monitor objects are like any other *SmartDB* objects; they observe the system evolution through their inputs, process this data through their state evolution, and output the desired statistics.

3.3 Process Architecture

In SmartDB, object state evolution is driven by passage of time or by occurrence of events. Events are generated by SmartDB objects as output messages and are communicated to the addressed objects as input messages. In this section we describe how the *SmartDB* architecture guarantees the timely evolution of each object and the timely communication of events to objects.

Time passage in *SmartDB* is represented by a global clock, which defines the smallest time step of the system. All evolution takes place at discrete advancements of this clock.

Each time driven object specifies the time step for its state evolution as a multiple of the global clock step. Rapidly evolving objects such as engines change their state more frequently, while more passive objects such as a roadside link controllers change their state at larger time steps.

Event driven objects exercise their behavior only when events are delivered to them. Coordination objects in vehicles, for example, respond to "Merge Request"s coming from other vehicles; the network layer changes the routing table upon an "accident".

Process layers are used to execute the simulation of collections of objects that evolve at same time steps. The process layers themselves can be time or event driven. Process layers are controlled by a "Process Coordinator". The process coordinator schedules the execution of time driven layers based on their time step and schedules the execution of event driven process layers if any events are raised against them.

A process layer can do the following:

- simulate the time driven evolution of all instances of an object type,
- for all instances of an object type with an outstanding event, deliver the event and simulate the event driven evolution.

If event driven objects are put in a time driven process layer, event delivery for these objects takes place only when the corresponding process layer is executed.

In a simulation run the process coordinator executes the process layers according to their time step, which in turn execute the simulation of the objects they contain.

The process architecture that implements the layered architecture proposed by Varaiya [7] is shown in Figure 6.

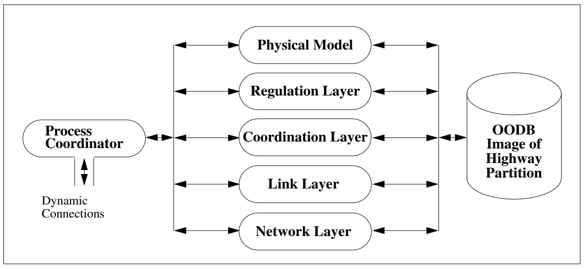

Figure 6: *SmartDB* Process Architecture.

The physical layer is time driven and simulates the time driven engine objects that generate the vehicle displacements.

The regulation layer is time driven. It contains event driven regulation supervisors and time driven maneuver objects. The supervisors switch between maneuvers based on incoming messages from the coordination layer; the maneuvers control the behavior of the gas pedal and the steering wheel.

The coordination layer is time driven. It contains event driven coordination objects. Coordination objects in different vehicles exchange messages to determine the maneuver a vehicle should execute. These decisions are communicated to regulation supervisors through messages.

The link layer is time driven and contains time driven link objects that set traffic parameters such as target speed and average platoon size in highway sections.

The network layer is event driven. It is executed only if an accident occurs. Upon an accident it reconfigures the routing tables.

The simulation objects are placed in an object-oriented database (OODB). The database provides a natural mechanism to save and store the state of a simulation.

3.4 Open and Distributed Architecture

SmartDB has an open architecture. *SmartDB* allows the user to create the desired simulation granularity by configuring a process layer architecture. For each process layer the user specifies the object types for time and/or event driven simulation. The OODB makes the simulation state visible to any user. It has a well-defined interface, and provides a default mechanism for any other simulation package to interface with *SmartDB*.

SmartDB supports distributed simulation. Zones serve as the unit of distribution: different zones can be distributed to different processors; they have their own database and their own clock. Since a vehicle can communicate with and sense other vehicles only in its neighborhood, communication between the distributed processors is restricted to objects in adjacent highway segments. Thus, locality of reference enables efficient distributed processing. Distributed simulation is depicted in figure 7. The boundary object between the last section of the previous zone and the the first section of the next zone coordinates communication between the processors, ensures synchronization of simulation clocks on different processors, and controls object migration between databases.

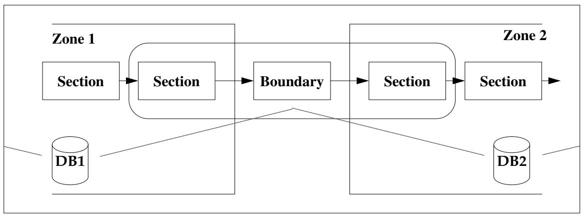

Figure 7: Distributed Simulation

4 Summary

SmartDB is a framework for uniform specification, simulation, optimization, evaluation, and implementation of IVHS alternatives.

SmartDB reduces system specification to mixing and matching of software components. These components are used to create highways, traffic patterns, and control and communication hierarchies.

A simulation run occurs when all objects in a given highway and traffic configuration execute their state evolution behavior.

The object behaviors are parametrized, and the performance of the transportation system being simulated can be optimized by adjusting these parameters based on specified optimization criteria.

The system performance can be observed and evaluated using monitor agents that collect statistical information about the system and generate performance reports. Different control alternatives are compared by simulating them with identical highway configurations and traffic patterns. The performance reports generated by the monitor agents in the respective simulations are used for objective comparison and evaluation of the alternatives.

Once a control architecture is selected, the individual software components can be implemented as physical hardware components for deployment .

References

[1] G. Booch. *Object Oriented Design with Applications.* Benjamin/Cummings, Redwood City, CA. 1991.

[2] P. Coad and E. Yourdon. *Object-Oriented Analysis.* Yourdon Press, Englewood Cliffs, NJ. 1991.

[3] F. Eskafi and P. Varaiya. Smartpath: Automatic Highway Simulator. *PATH Technical Memorandum*, UC Berkeley. June 1992.

[4] IVHS America. *Strategic Plan for Intelligent Vehicle-Highway Systems in the United States.* Report N0. IVHS-AMER-92-3. 20 May 1992.

[5] R. Jonson. How to Develop Frameworks. *OOPSLA Tutorial Notes.* ACM Press. 1993

[6] S. Shladover *et. al.* Automated Vehicle Control Developments in the PATH program. *IEEE Trans. Vehicular Tech.* Vol. 40. pp. 114-130. Feb. 1991.

[7] P. Varaiya. Smart Cars on Smart Roads: Problems of Control. *IEEE Trans. Automatic Control* Vol. 38, No 2. Feb. 1993.

[8] A. Göllü, Pravin Varaiya; Hybrid Dynamical Systems; In *Proceedings of the 28th Conference on Decision and Control,* pages 2708 2712, Tampa FL, December 1989.