TRANSIMS-Seminar

Integration of High and Low Fidelity Approaches in TRANSIMS

July 21st, 1994

Marcus Rickert

High Fidelity

- Intelligent agents
- Vehicle oriented storage in abstract data structures
- Numerous precise rules describing driver's behaviour
- Wide variety of parameters for network, vehicles, and drivers
- Currently time step driven, but goal: event driven system (?)
- Slow (?!)

Low fidelity

- Cellular automata
- Site oriented storage in arrays
- Few simple rules
- Hardly any parameters
- Time step driven
- Fast!

Goals

- As many common data structures for HF and LF as possible

- High computational speed

- Distributed system

- Dynamic load balancing

- Individual choice of underlying model for each segment

- Modular structure

- Independence of hardware

Hardware

- Workstation clusters
  - Inhomogenous performance
  - Inhomogenous binary representation of data (int, double)
  - Non-constant number of CPNs
  - Low communication throughput
  - High latency

Sparc or IBM-Risc clusters

Hardware (cont'd)

- Parallel computer systems
  - Identical CPNs
  - High communication throughput
  - Low latency

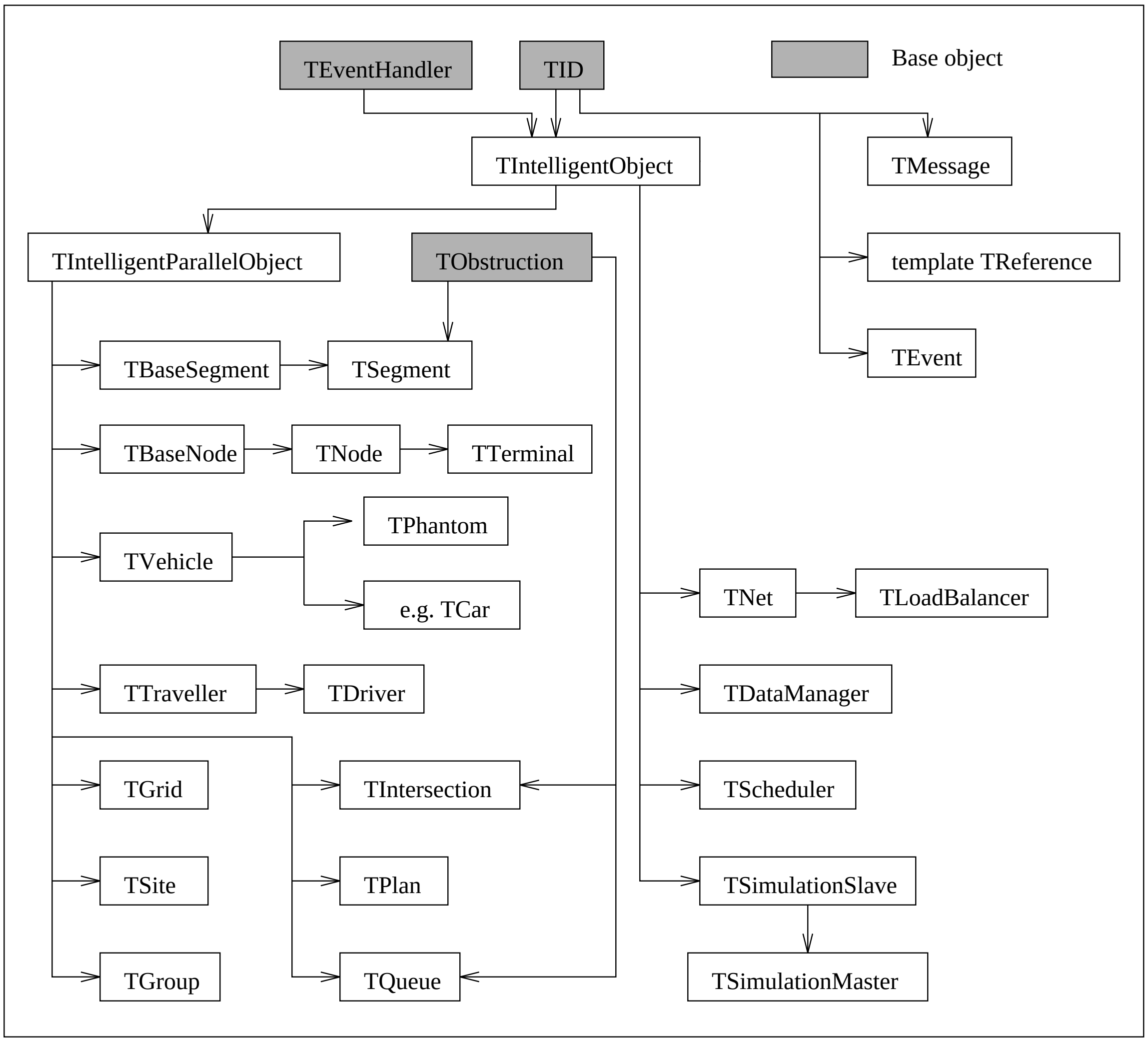
CM-5
T3D
Paragon
Parsytec

Software

- C++ object oriented (strict for HF, relaxed for LF)
  - inheritence
  - encapsulation
  - virtual methods

- PVM
  - Message passing
  - CPN control

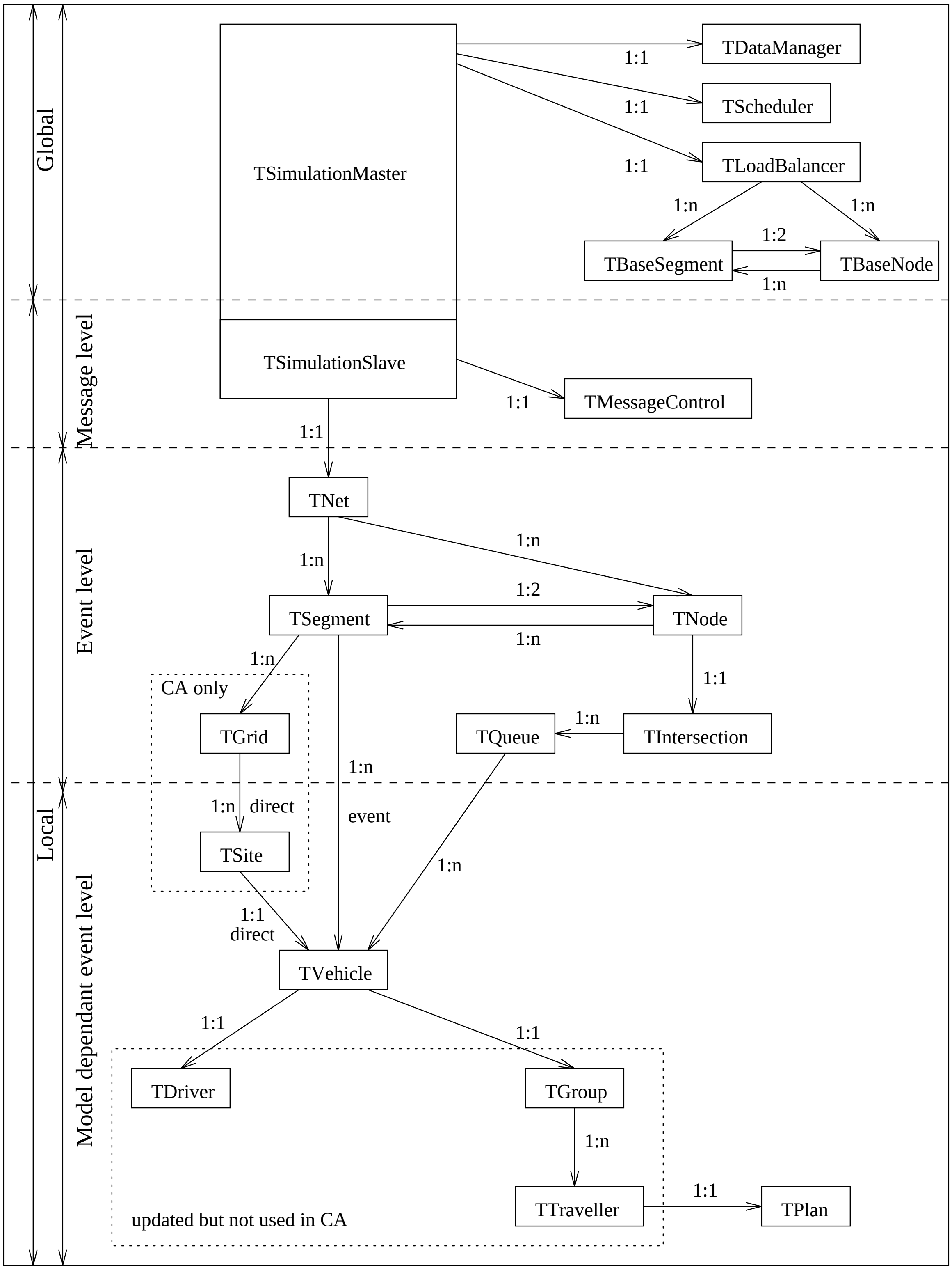
General availability



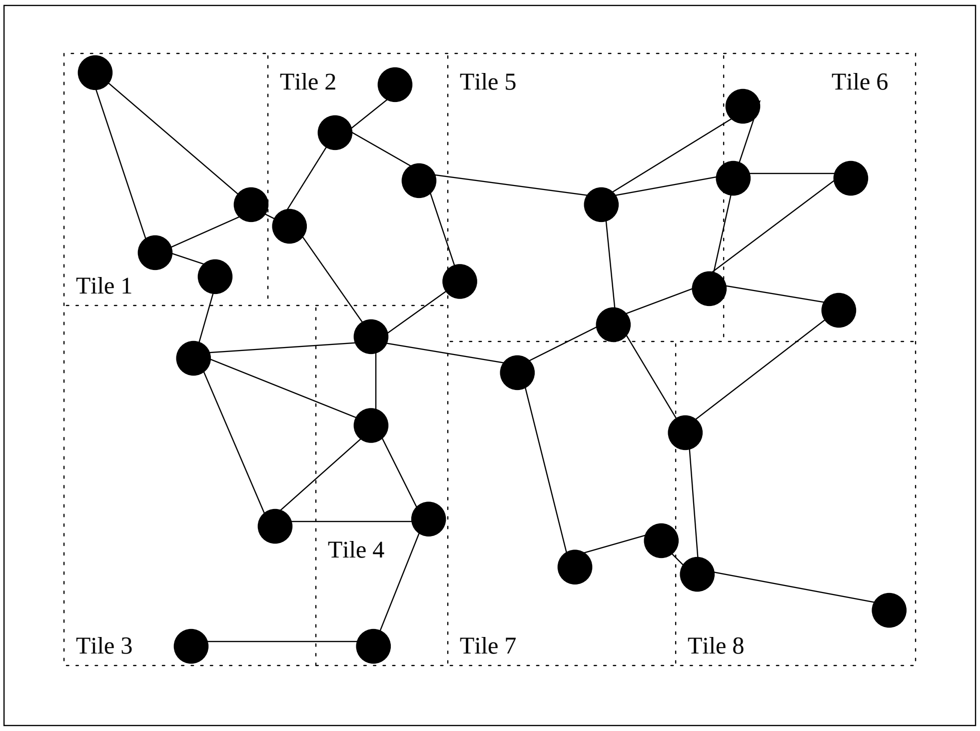
TEventHandler
TID
Base object
TIntelligentObject
TMessage
TIntelligentParallelObject
TObstruction
template TReference
TBaseSegment
TSegment
TEvent
TBaseNode
TNode
TTerminal
TPhantom
TVehicle
e.g. TCar
TNet
TLoadBalancer
TTraveller
TDriver
TDataManager
TGrid
TIntersection
TScheduler
TSite
TPlan
TSimulationSlave
TGroup
TQueue
TSimulationMaster

Parallel structure

- Geometric distribution in tiles
- Tiles as konvex as possible
- As few boundaries as possible
- SIMD code
- One master CPN, multiple slave CPNs
- Message driven



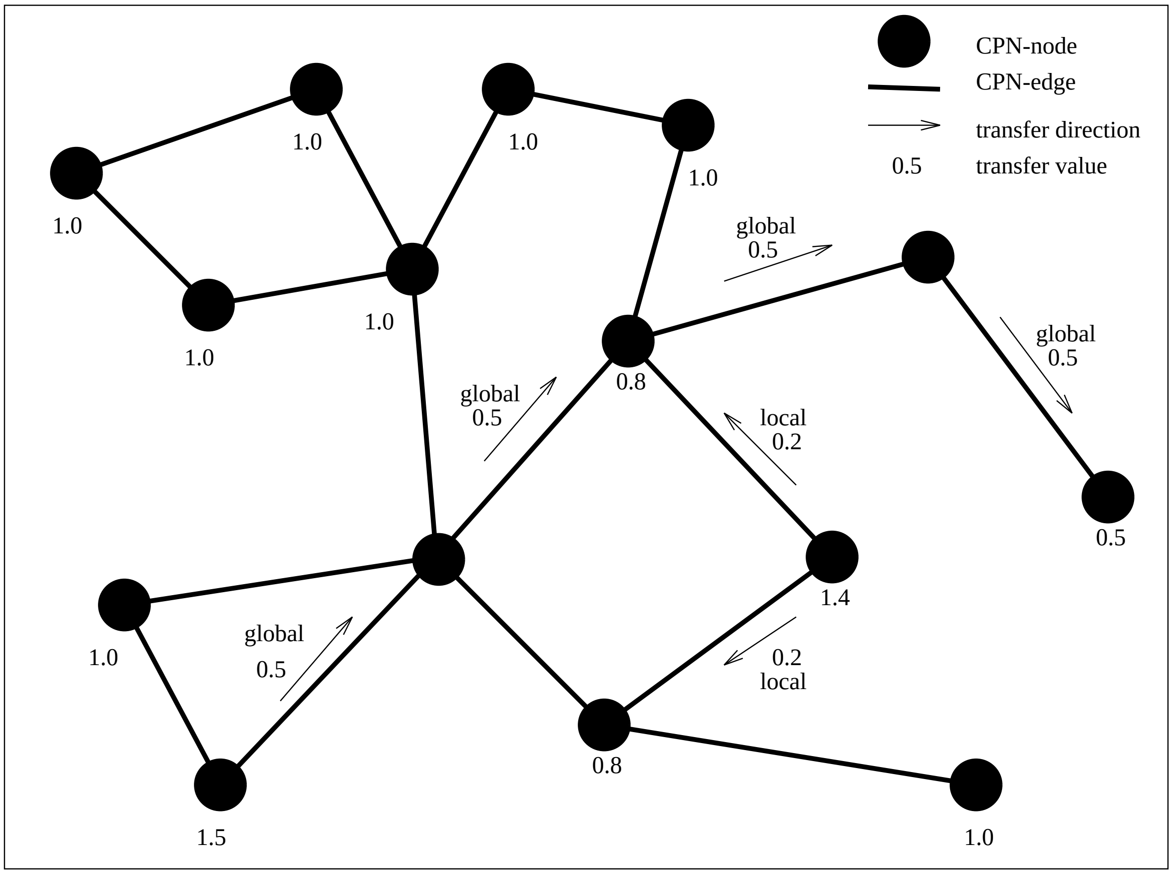
Global
Message level
Event level
Local
Model dependant event level
TSimulationMaster
TDataManager
TScheduler
TLoadBalancer
1:1
1:1
1:1
1:n
1:n
TBaseSegment
TBaseNode
1:2
1:n
TSimulationSlave
TMessageControl
1:1
1:1
TNet
1:n
1:n
TSegment
TNode
1:2
1:n
1:n
CA only
TGrid
1:1
TQueue
TIntersection
1:n
1:n
1:n
direct
event
TSite
1:n
1:1
direct
TVehicle
1:1
1:1
TDriver
TGroup
1:n
TTraveller
1:1
TPlan
updated but not used in CA



Tile 1
Tile 2
Tile 3
Tile 4
Tile 5
Tile 6
Tile 7
Tile 8

Load balancing

- Local
  - Distribute load surplus onto neighbouring nodes
  - May result in load gradients
  - Local implementation possible
- Global
  - Find CPN with greatest load
  - Find Nodes with load deficiency
  - Transfer load on shortest paths
  - Can resolve gradients
  - Local implementation impossible/difficult?



CPN-node
CPN-edge
transfer direction
0.5
transfer value
1.0
1.0
1.0
1.0
1.0
1.0
1.0
global
0.5
global
0.5
0.8
global
0.5
local
0.2
0.5
global
0.5
1.4
1.0
0.2
local
0.8
1.5
1.0

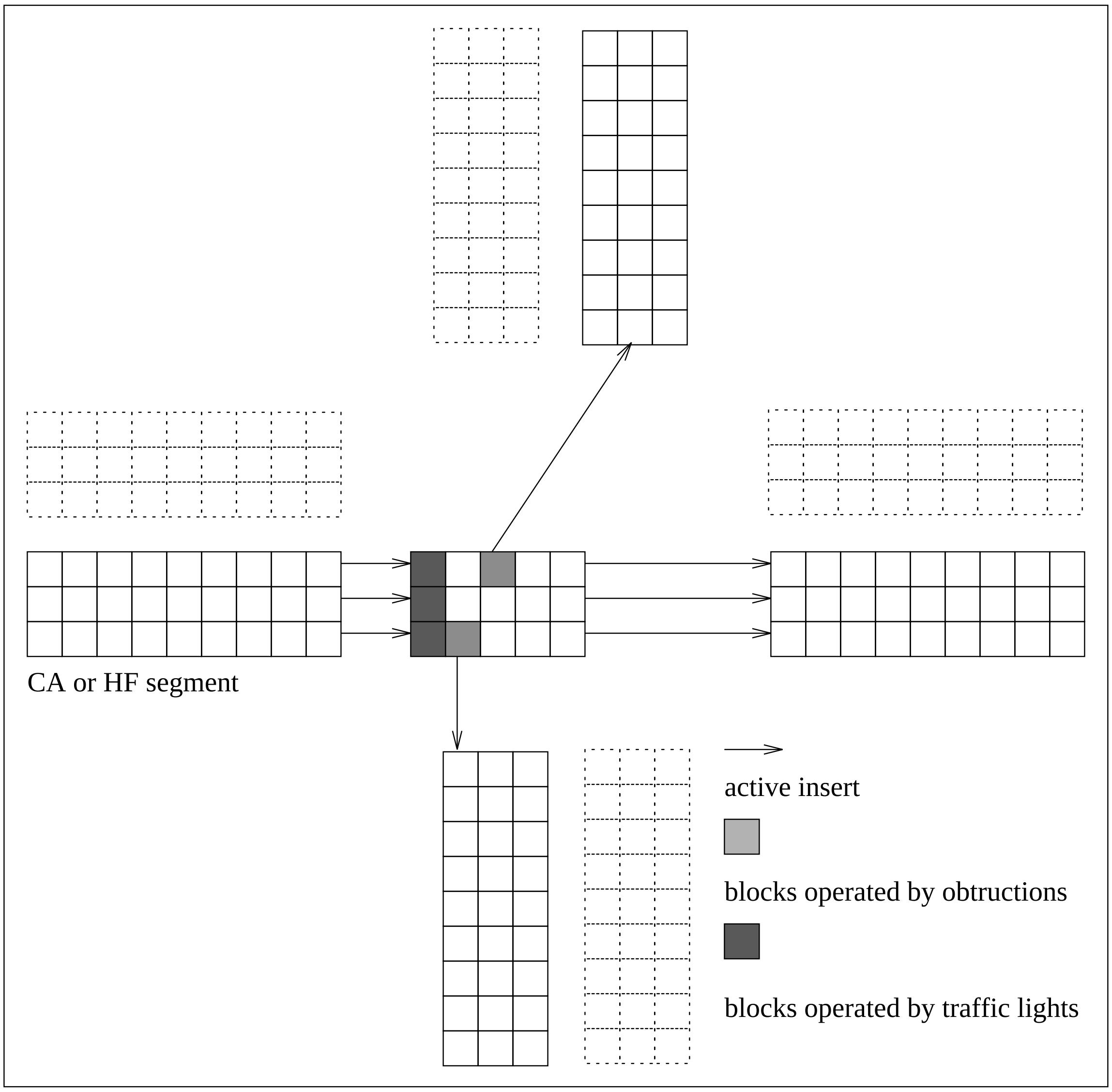
Intersections

- Serve as interfaces between HF and LF

- Should be simple and fast in order not to slow down LF

- Should be 'invisible' for through lanes in HF

- Capable of blocking turning/through lanes due to
  - other lanes
  - incedents


CA or HF segment
active insert
blocks operated by obtructions
blocks operated by traffic lights

Handling of vehicles

How are vehicles handled on a segment? Re-sort them in every time step or keep a sorted data structure?

- AB-tree
- Doubly linked list
- Grid based storage