Near Real-Time Distributed Enterprise Infrastructure for Traffic Data Collection and Dissemination Using Java Technology

Brian Smyth, VP Software Development, NCS
Jim Carroll, Chief Software Architect, NCS

TS-6028

NAVTEQ®

Java

Visit: www.NN4D.com

Sun microsystems

To learn some of the techniques used in building a large scale near-real-time traffic analysis engine and with some example integrations of our data feeds

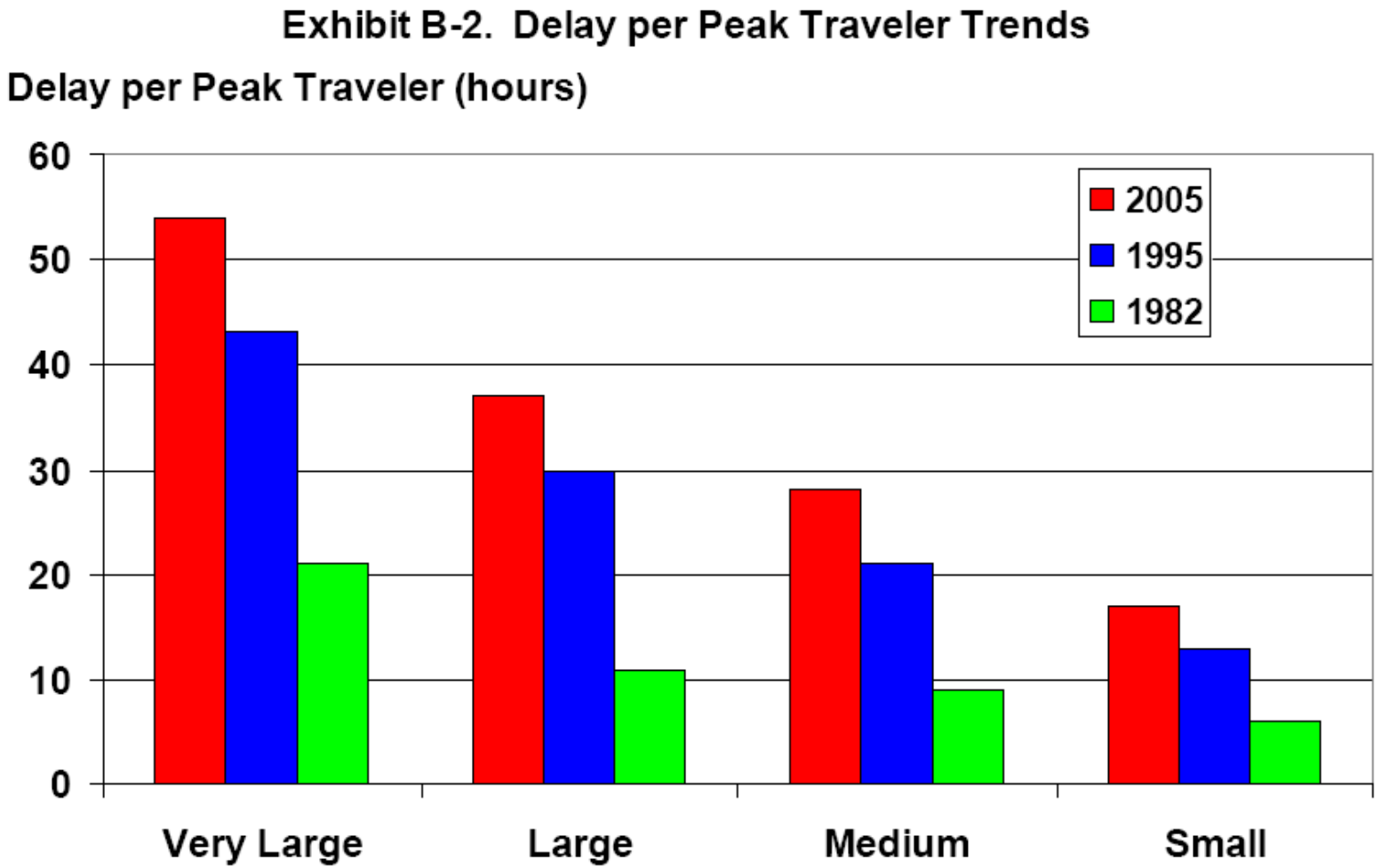
GOAL

Visit: www.NN4D.com

Agenda

- Introduction
- Data Collection
- Data Fusion/Middleware
- Data Dissemination
- Examples of Integrating our Traffic Feeds
- Questions

TTI 2005 Urban Mobility Report



Visit: www.NN4D.com

Advertisers Shift To Multi-Platform Integrated Campaigns

One-to-Many → *Personalization* → **One-to-One**

Broadcast

Best product placement in local news/traffic segments

TV

Sponsorship of traffic reports with virtual city 3D fly-throughs.

Radio

:10 sponsorships tied to 'active' listening content encourages high-retention

Internet

Measurable, growing audience with location awareness.

Traffic.com site

TrafficOne Network Sites:

radio, television, newspaper, co-branded traffic sites

Wireless

Message geo-targeted through alerts

Outbound phone alerts with direct connect & inbound **1-866-MY-TRAFC** traffic hotline.

Text and HTML email alerts; 42% open rate (5x industry standard)

Visit: www.NN4D.com

Traffic Content Creation / Management

- Data is collected from a variety of sources and fused together
  - Fixed point sensors: Provide lane-by-lane detail of every vehicle from our own and DOT sensors in more then 30 cities
    - 40MM+ records/day
  - GPS probe: provides location and speed data on an individual vehicle across all major highways and arterials
    - 20MM+ records/day and growing rapidly
  - Incident data: accidents, construction, events, congestion, etc
    - 450+ employees collecting more then 40,000 records/day

Traffic.com/DOT Sensor Network

Probe Data

Incident and Event Data

Distribution and Products

- NAVTEQ is the only traffic provider with ability to directly deliver traffic content across all platforms
  - Broadcast Data feeds (Satellite, RDS and HD)
  - Internet
  - Mobile (SMS, MMS, mobile video)
  - Telephone (interactive voice response)
  - TV and Radio Broadcast
- NAVTEQ provides the largest number of traffic turnkey solutions, not just data feeds
  - Web Services (traffic.com and TrafficOne solutions)
  - Mobile Applications (traffic.mobi.com)
  - RDS and HD
  - Television and Radio broadcast applications (NeXgen and Broadcaster)
  - Mobile video solutions (JamCast)

No other traffic company takes their services to the last mile with complete solutions for customers

Visit: www.NN4D.com

Television Solutions Platforms

- **NeXgen™ Platform**
  - Customizable, integrated television graphics system for display of real-time traffic flow, incidents, travel times
  - 2-D, 3-D Skyview, and 3-D City Fly-through formats engage viewers
  - Sponsorships provide first in or last out of commercial sets within traffic, news or weather programming
- **JamCast™ Platform**
  - 24/7, dynamic traffic reporting service for digital tier, web, and wireless
  - Delivers television branding on-air, online and on-the-go

Protected under US Patent 7,116,326

Visit: www.NN4D.com

Radio Solutions Platforms

- **Broadcaster™ Platform**
  - Web-based dashboard for automatic broadcast script generation
  - Provides extensive reach for national, multi-market, or local radio campaigns
- **RDS & HD Radio® Platform**
  - Real-time traffic content supplied for digital radio channels
  - Extends advertising reach with newest technologies
- **Satellite Radio Platform**
  - Maps, traffic and location content delivered via XM and Sirius
  - Delivers on-the-road map search listings for advertisers

Visit: www.NN4D.com

Online Solutions Platforms

- **Traffic.com Platform**
  - Real-time traffic anytime, anywhere
  - MyTraffic™ for personalized alerts delivered exactly when, where and how users request
  - Web, email, voice, video, WAP and SMS advertising sponsorships
- **TrafficOne™ Platform**
  - Traffic.com network of co-branded web sites and MyTraffic users
  - Extended reach across 200+ local, news, directory and travel sites
- **JamCast™ Platform**
  - Automated, digital 24/7 real-time traffic video designed to engage the viewer
  - Embedded pre- and post-roll promotions

Visit: www.NN4D.com

Traffic.com *Wireless*

Delivering real-time, actionable traffic information however wireless users want it

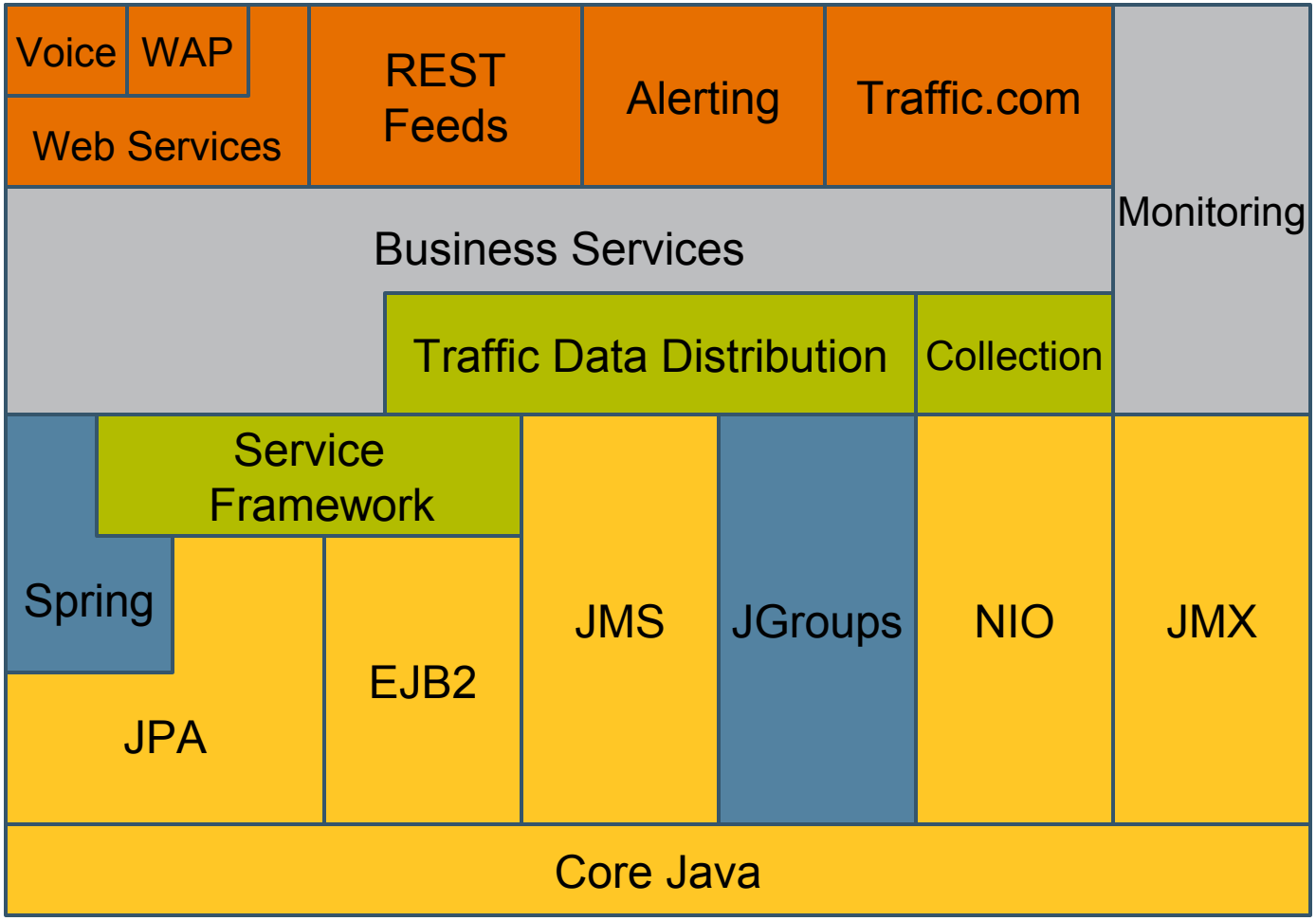
SMS	Voice	WAP	JamCast™	MMS	J2ME
A standard or premium messaging service that supports the pull of traffic summaries, metro problem areas and particular roads as well as push of traffic alerts based on time and user defined thresholds	A free or premium voice based service in which the user can call and receive real time traffic information, travel times and detailed incident reports at any time.	A multi function WAP solution mirroring functionality of the current web site offered in both premium or ad supported models	Audio and video traffic solution delivering real-time traffic report information in 2D, 3D and audio, on streaming and downloadable media.	A solution combining the immediacy of wireless messaging services with rich text, audio, graphic and video information	A rich media, wireless solution mirroring functionality of the current web site, providing both summary and route specific traffic information in a clear and easy to deliver format.

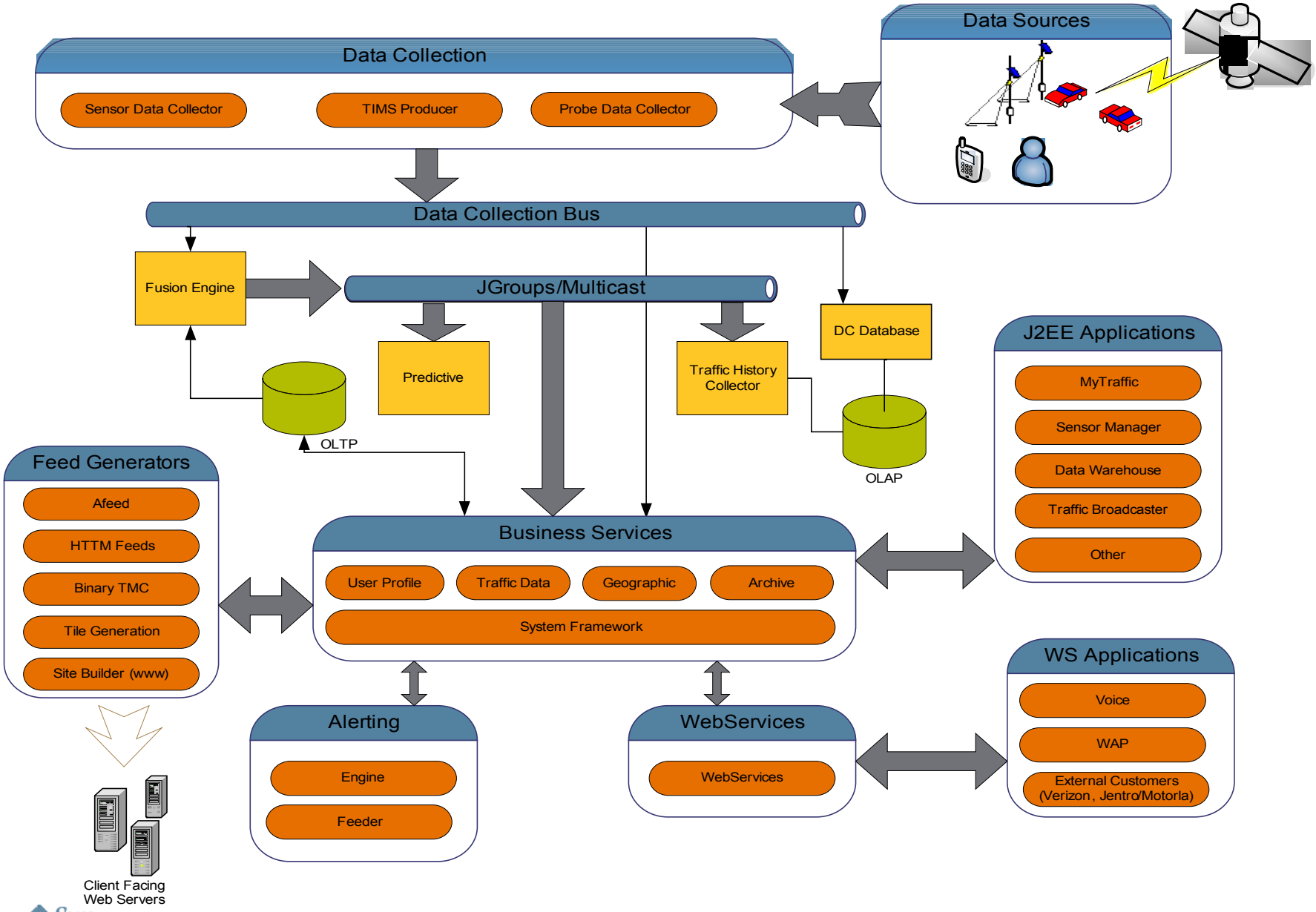
Visit: www.NN4D.com

Market Leaders Across the Value Chain Use NAVTEQ Traffic

Visit: www.NN4D.com

Technology Stack

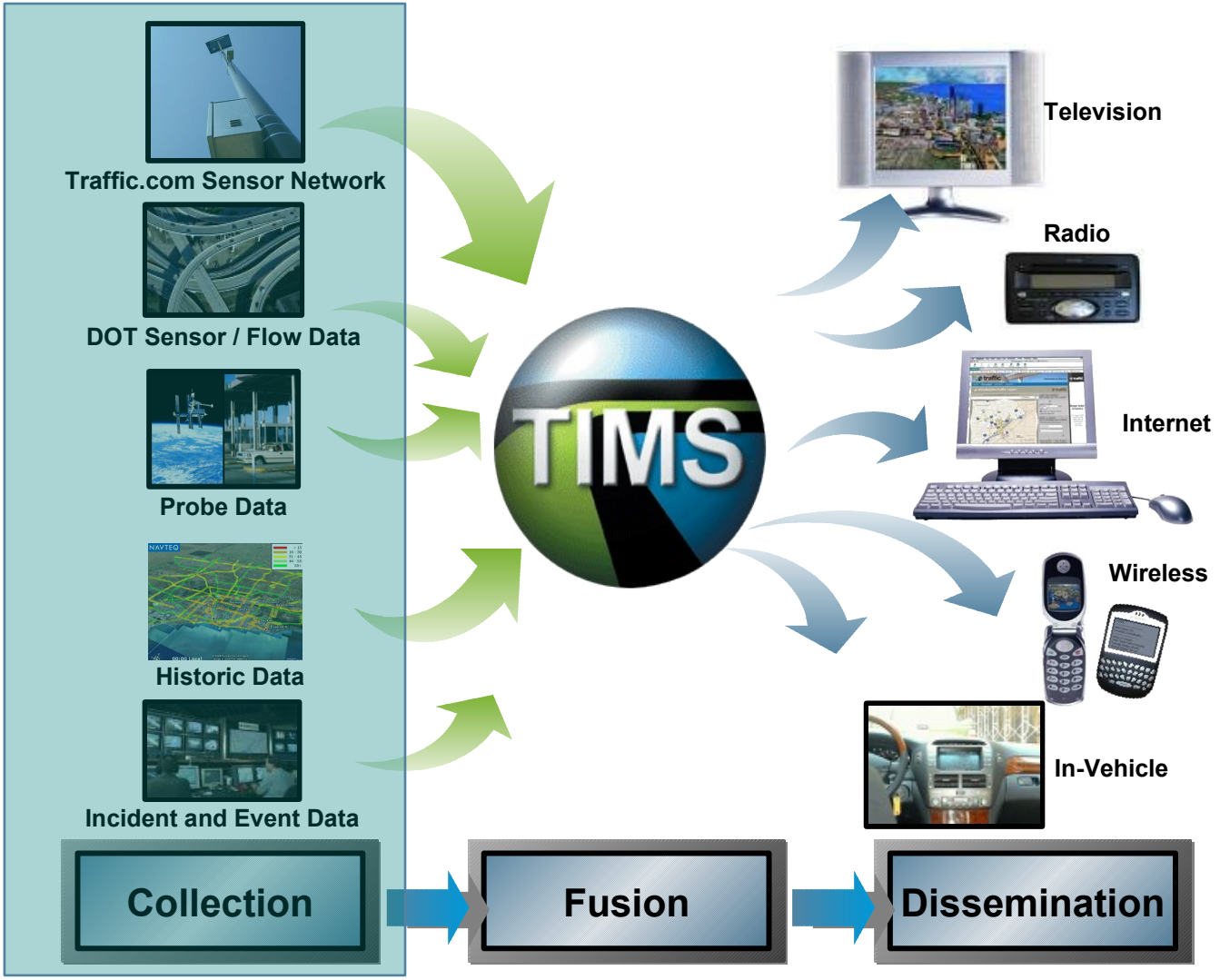



Data Sources
Data Collection
Sensor Data Collector
TIMS Producer
Probe Data Collector
Data Collection Bus
Fusion Engine
JGroups/Multicast
DC Database
Predictive
Traffic History Collector
OLTP
OLAP
J2EE Applications
MyTraffic
Sensor Manager
Data Warehouse
Traffic Broadcaster
Other
Feed Generators
Afeed
HTTM Feeds
Binary TMC
Tile Generation
Site Builder (www)
Business Services
User Profile
Traffic Data
Geographic
Archive
System Framework
WS Applications
Voice
WAP
External Customers (Verizon, Jentro/Motorla)
Alerting
Engine
Feeder
WebServices
WebServices
Client Facing Web Servers

Agenda

- Introduction
- Data Collection
- Data Fusion/Middleware
- Data Dissemination
- Examples of Integrating our Traffic Feeds
- Questions

End-to-End Solution



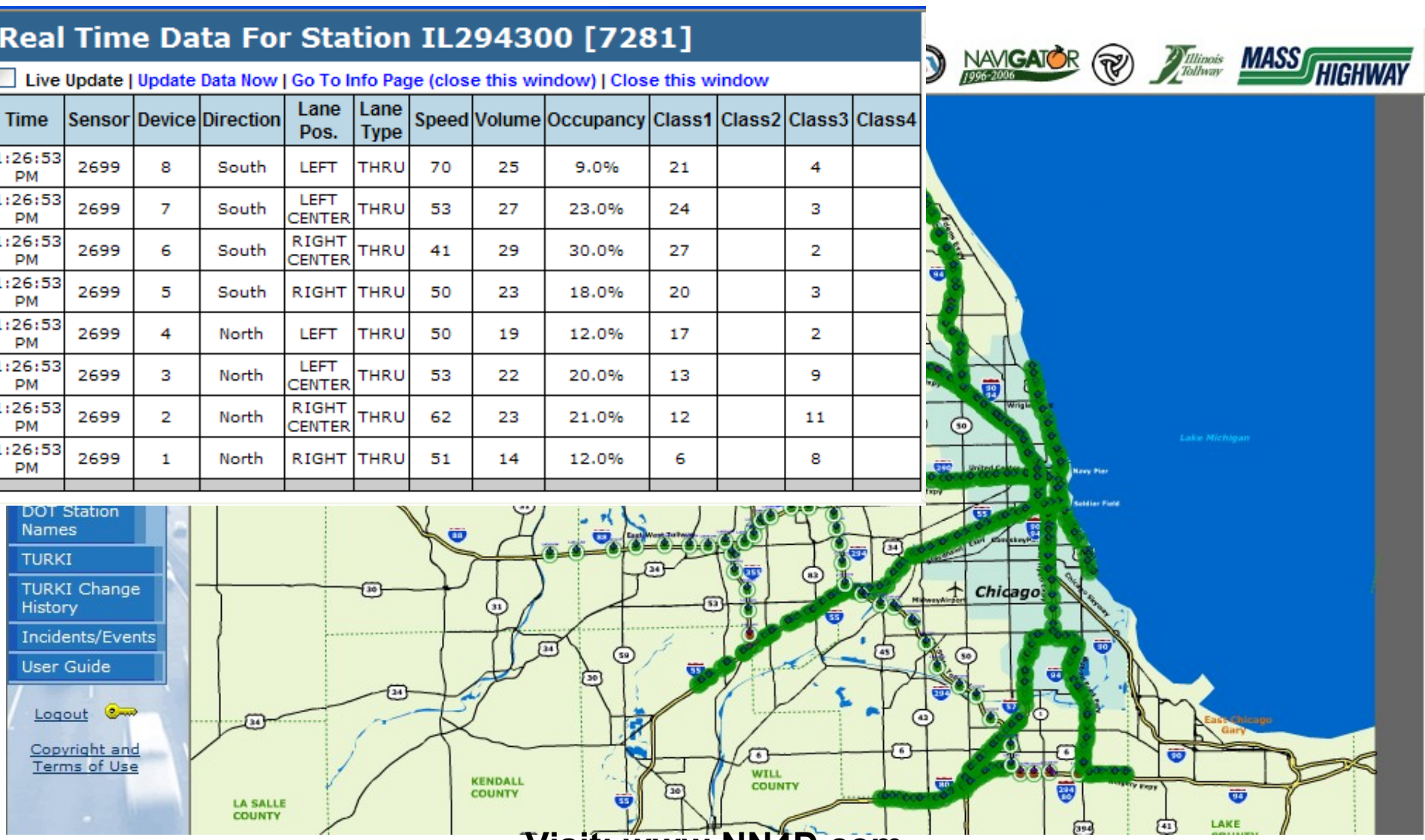
Visit: www.NN4D.com

Sensor Data Collection

Real Time Data For Station IL294300 [7281]

Live Update | Update Data Now | Go To Info Page (close this window) | Close this window

Time	Sensor	Device	Direction	Lane Pos.	Lane Type	Speed	Volume	Occupancy	Class1	Class2	Class3	Class4
1:26:53 PM	2699	8	South	LEFT	THRU	70	25	9.0%	21		4	
1:26:53 PM	2699	7	South	LEFT CENTER	THRU	53	27	23.0%	24		3	
1:26:53 PM	2699	6	South	RIGHT CENTER	THRU	41	29	30.0%	27		2	
1:26:53 PM	2699	5	South	RIGHT	THRU	50	23	18.0%	20		3	
1:26:53 PM	2699	4	North	LEFT	THRU	50	19	12.0%	17		2	
1:26:53 PM	2699	3	North	LEFT CENTER	THRU	53	22	20.0%	13		9	
1:26:53 PM	2699	2	North	RIGHT CENTER	THRU	62	23	21.0%	12		11	
1:26:53 PM	2699	1	North	RIGHT	THRU	51	14	12.0%	6		8	



Visit: www.NN4D.com

Sensor Data Collection Considerations

- Thousands of individual sensors
- Dozens of communication methods and sensor types
- 40 million+ sensor records per day
- Keep distribution of data as close to real-time as possible

Sensor Data Collection Design Decisions

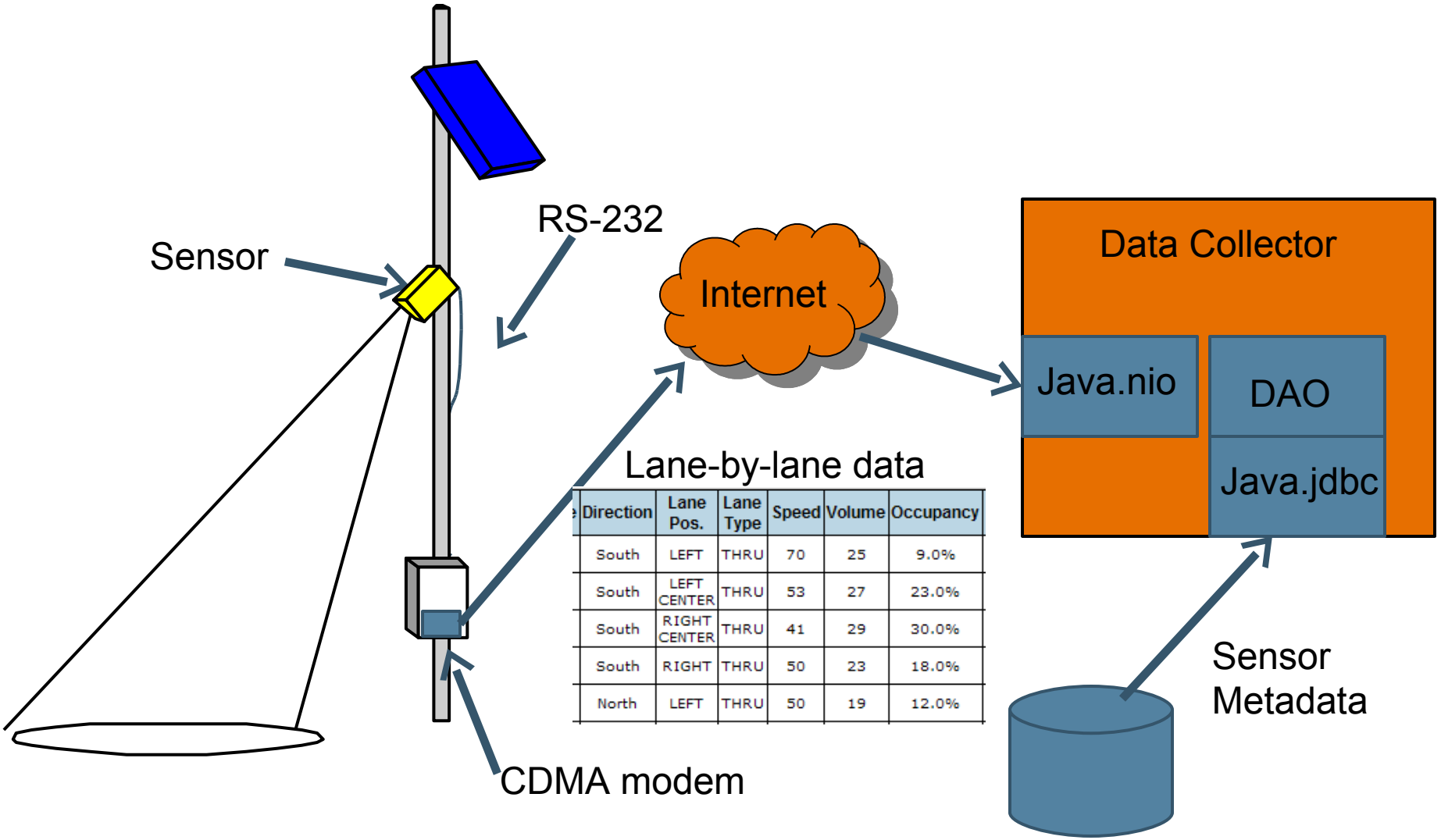
- Standard use of java.io uses blocking i/o
- Blocking i/o requires a thread per connection
- Context switching degrades performance noticeably starting at about 500 threads

Sensor Data Collection Design Decisions

- Standard use of java.io uses blocking i/o
- Blocking i/o requires a thread per connection
- Context switching degrades performance noticeably starting at about 500 threads

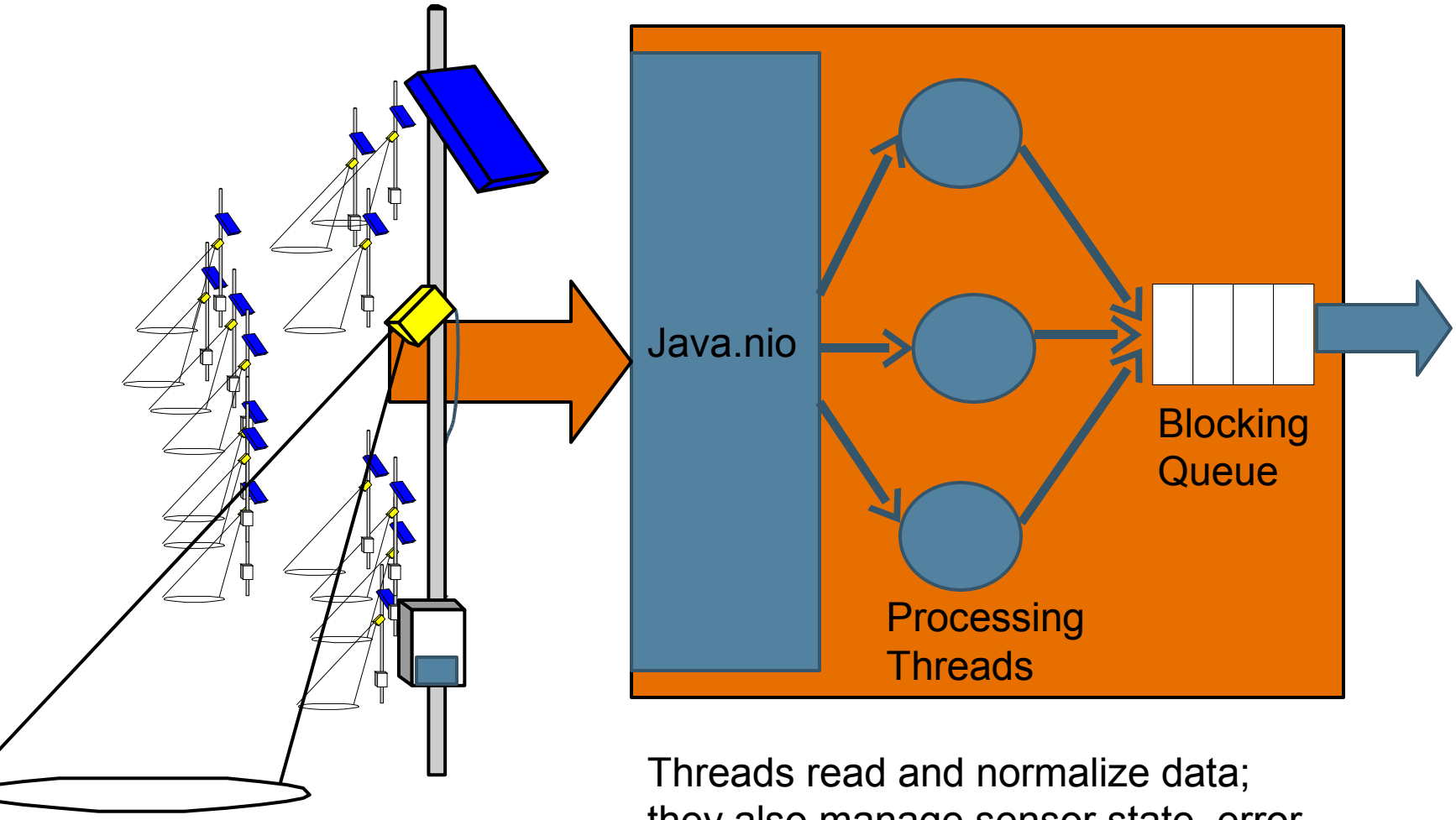
- Java™ New I/O API
  - java.nio allows us to control multiple connections from a single thread
  - We can adjust the thread/connection ratio for optimal performance
- Normalization of sensor data

Data Collection



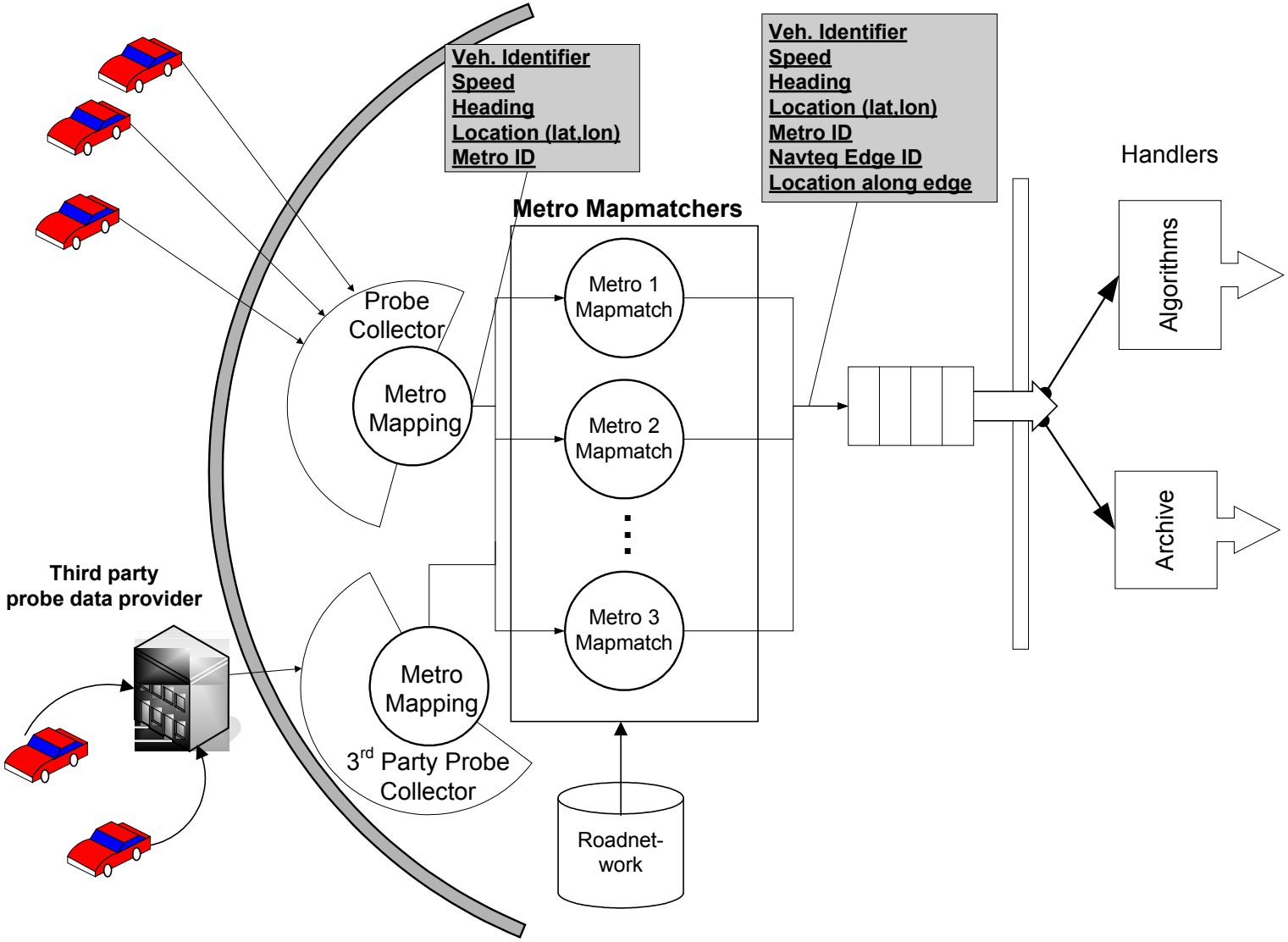
Direction	Lane Pos.	Lane Type	Speed	Volume	Occupancy
South	LEFT	THRU	70	25	9.0%
South	LEFT CENTER	THRU	53	27	23.0%
South	RIGHT CENTER	THRU	41	29	30.0%
South	RIGHT	THRU	50	23	18.0%
North	LEFT	THRU	50	19	12.0%

Data Collection and Dissemination



Threads read and normalize data; they also manage sensor state, error conditions and alerts to operations staff

Probe Data Collection

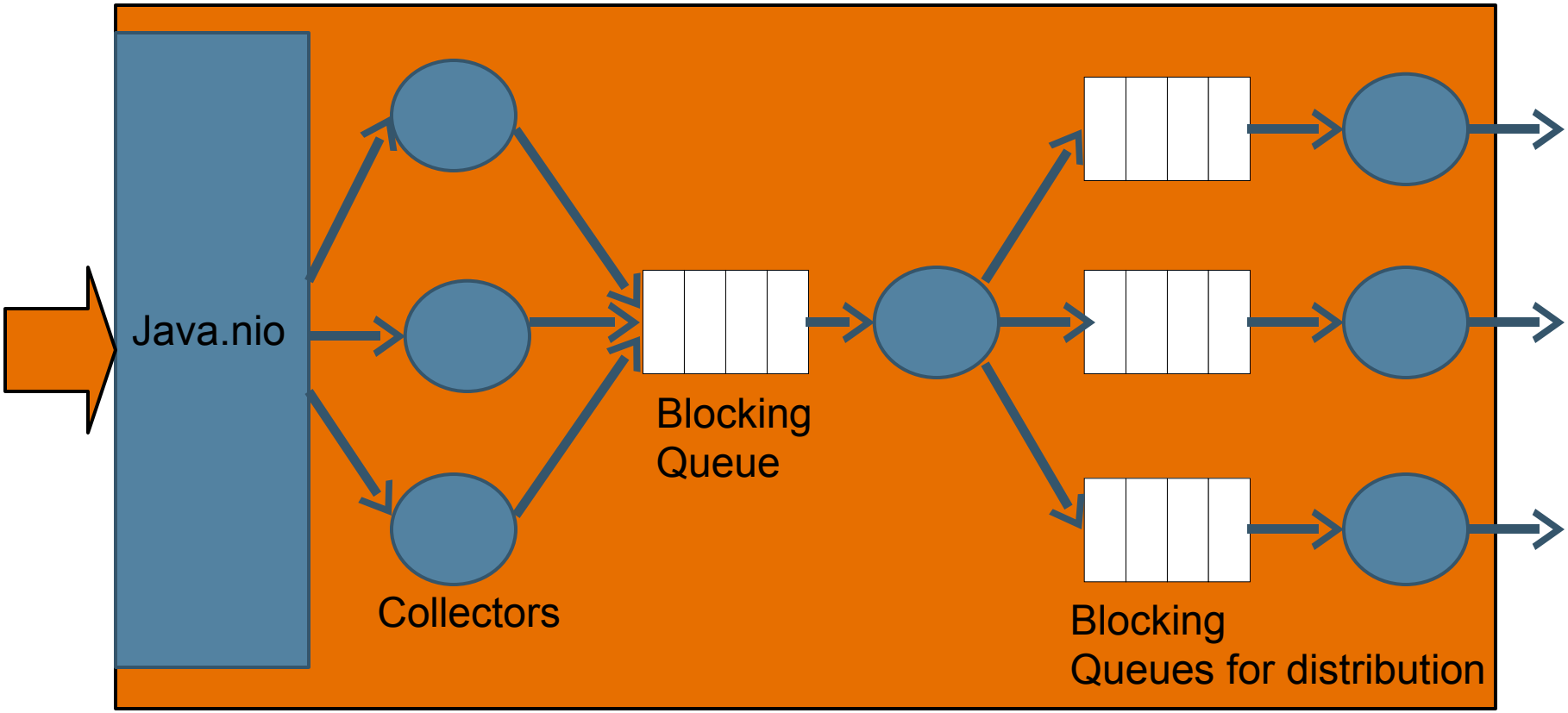


Data Collection Distribution Concerns

- Publish/Subscribe
- 60 Million+ records need to be distributed
- Latency needs to be minimized

- Asynchronous Archiving
- Custom Serialization for Compression
- Geographically isolated interest

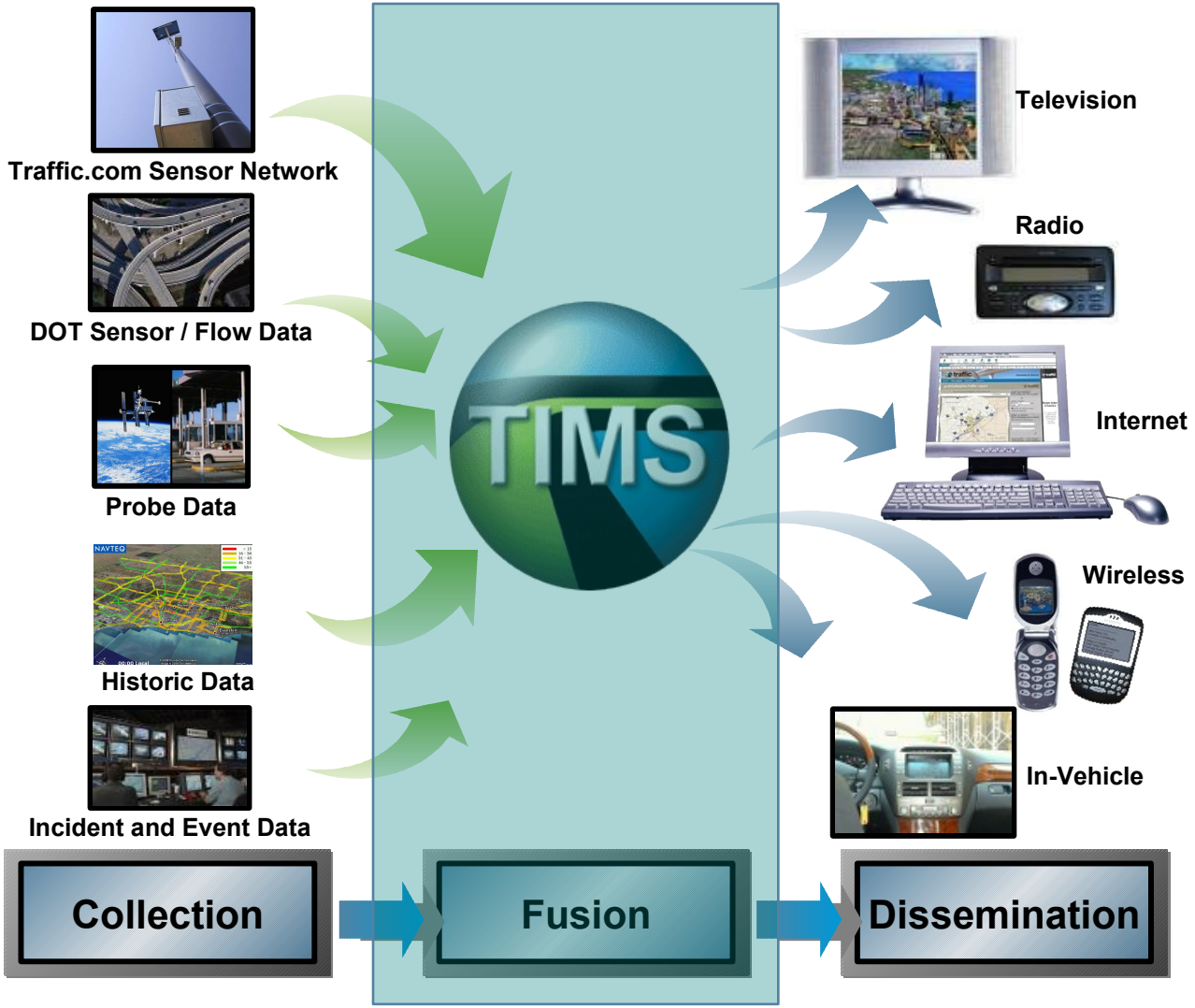
Data Collection and Dissemination



Agenda

- **Introduction**
- Data Collection
- Data Fusion/Middleware
- Data Dissemination
- Examples of Integrating our Traffic Feeds
- Questions

End-to-End Solution



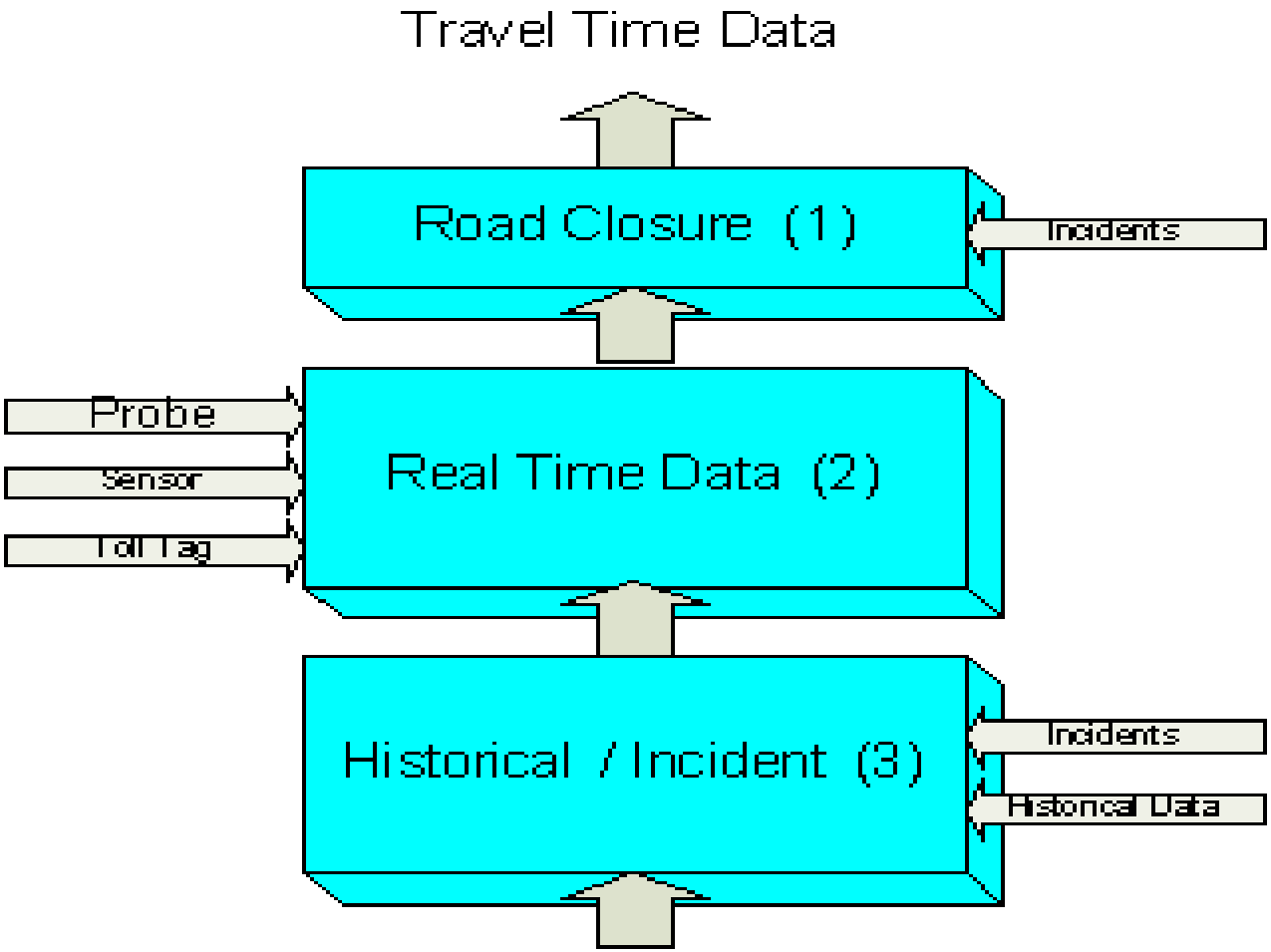
Visit: www.NN4D.com

Fusion Engine

- Hierarchical Travel Time Model (HTTM)
- Incorporate all data sources
  - Fixed Sensors
  - Toll tags/Link speeds
  - Vehicle Probes (GPS Devices)
  - Probe Sequences (previous probe combined with current probe)
  - Incident Data (Accidents, Weather Events, Construction, etc.)
  - Historical Data
- Weighed average of data sources on a continuous basis
  - Weight decays with distance
  - Weight decays with time
- Confidence calculation
- 1,000,000 road miles in U.S.

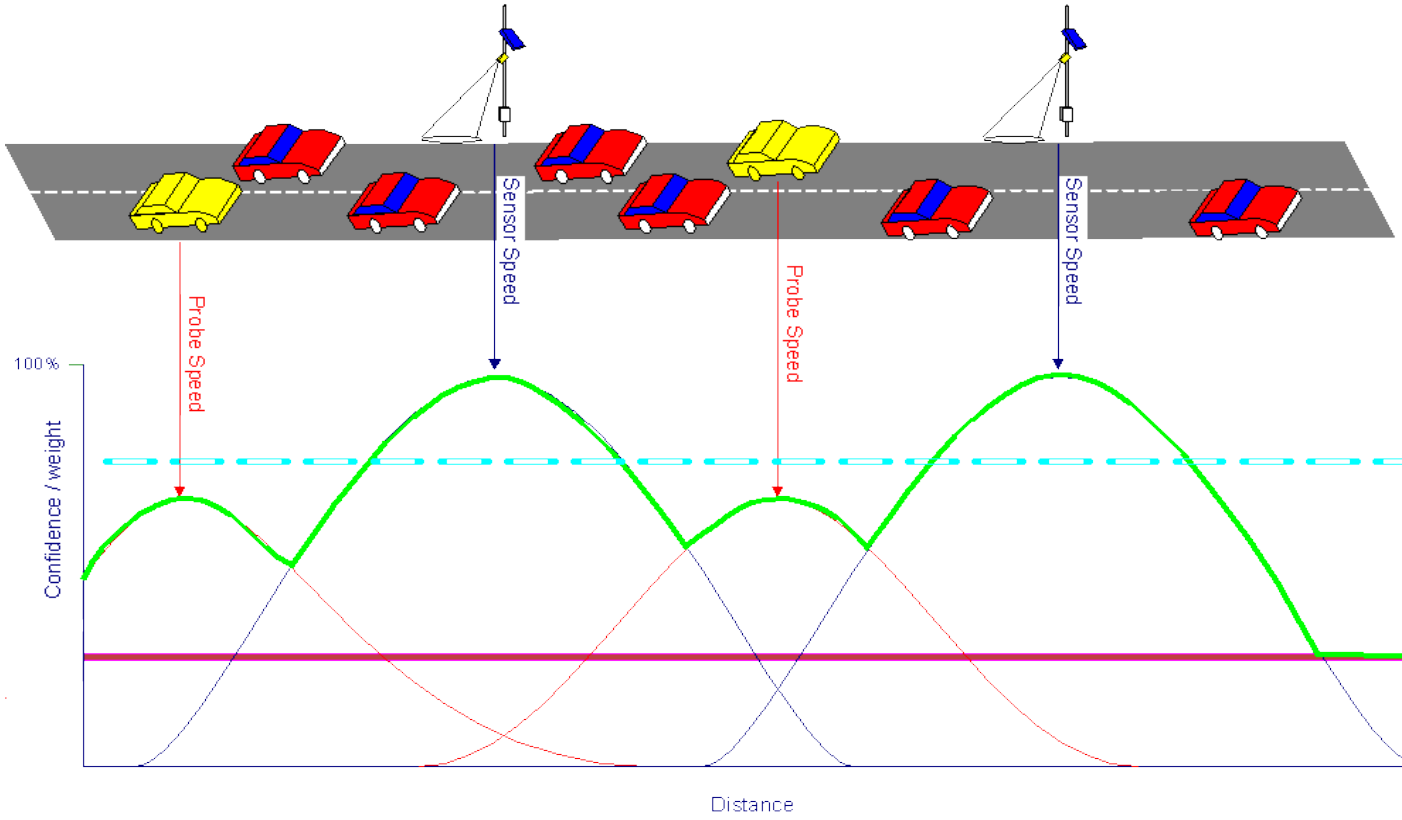
Visit: www.NN4D.com

Travel Time Data Hierarchy Refines Travel Times



Visit: www.NN4D.com

HTTM Algorithm



Legend

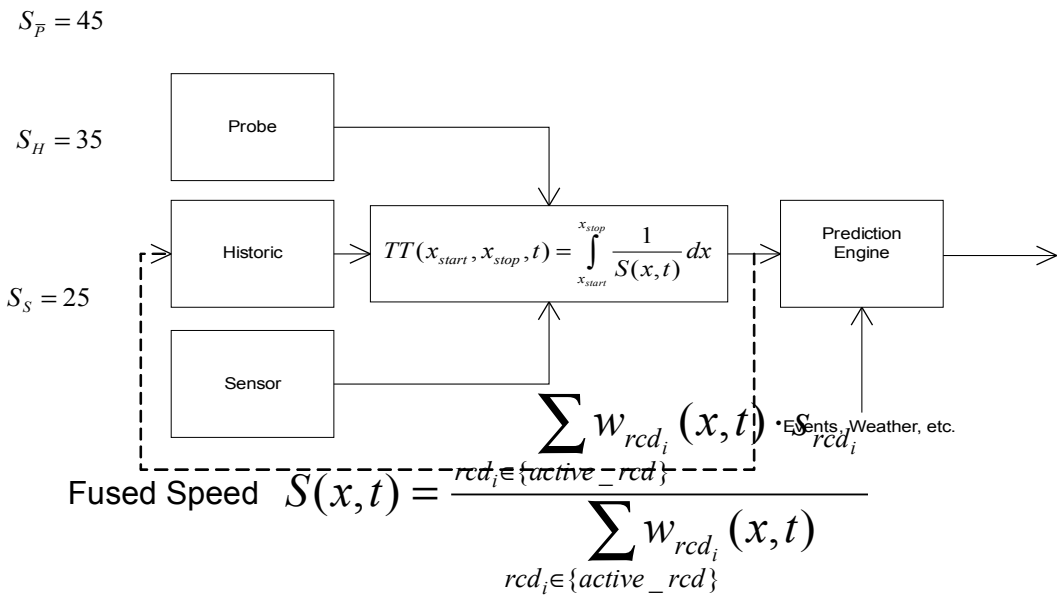
- Effective confidence
- Historical confidence
- Actual confidence
- Probe weight
- Sensor weight

The data is derived from sensors, probe and historical values. When live data (sensor and probe) is available, they are combined by using weighted average of sensors' readings are numerically integrated across a path. The remainder of the route is calculated with Historical data. Actual confidence is maintained as the maximum confidence of any single value. Effective confidence is the aggregation of the Actual confidence.

Visit: www.NN4D.com

Data Fusion Business Logic

Different roadway condition data sources are fused by the engine



- Weight function is defined as following: $w_{rcd_i}(x,t) = w_{dist}(x, w_{time}(t, w_{init}))$

$$w_{dist}(x, w_{time}) = \begin{cases} 0, x < x_0 - \min\left(d_{\max}, \frac{d_p}{2}\right) \\ w_{time}, x_0 - \min\left(d_{\max}, \frac{d_p}{2}\right) \le x \le x_0 + \min\left(d_{\max}, \frac{d_n}{2}\right) \\ 0, x > x_0 + \min\left(d_{\max}, \frac{d_n}{2}\right) \end{cases}$$

$$w_{time}(t, w_{init}) = \begin{cases} 0, t < t_0 \\ w_{init} \cdot \left(1 - \frac{t - t_0}{t_{lifetime}}\right), t_0 \le t \le t_0 + t_{lifetime} \\ 0, t > t_0 + t_{lifetime} \end{cases}$$

Data Fusion Considerations

- Considerations
  - 1,000,000 road miles covered
  - Integration estimated over 1/40th of a mile
  - 40 Million calculations to obtain a complete picture

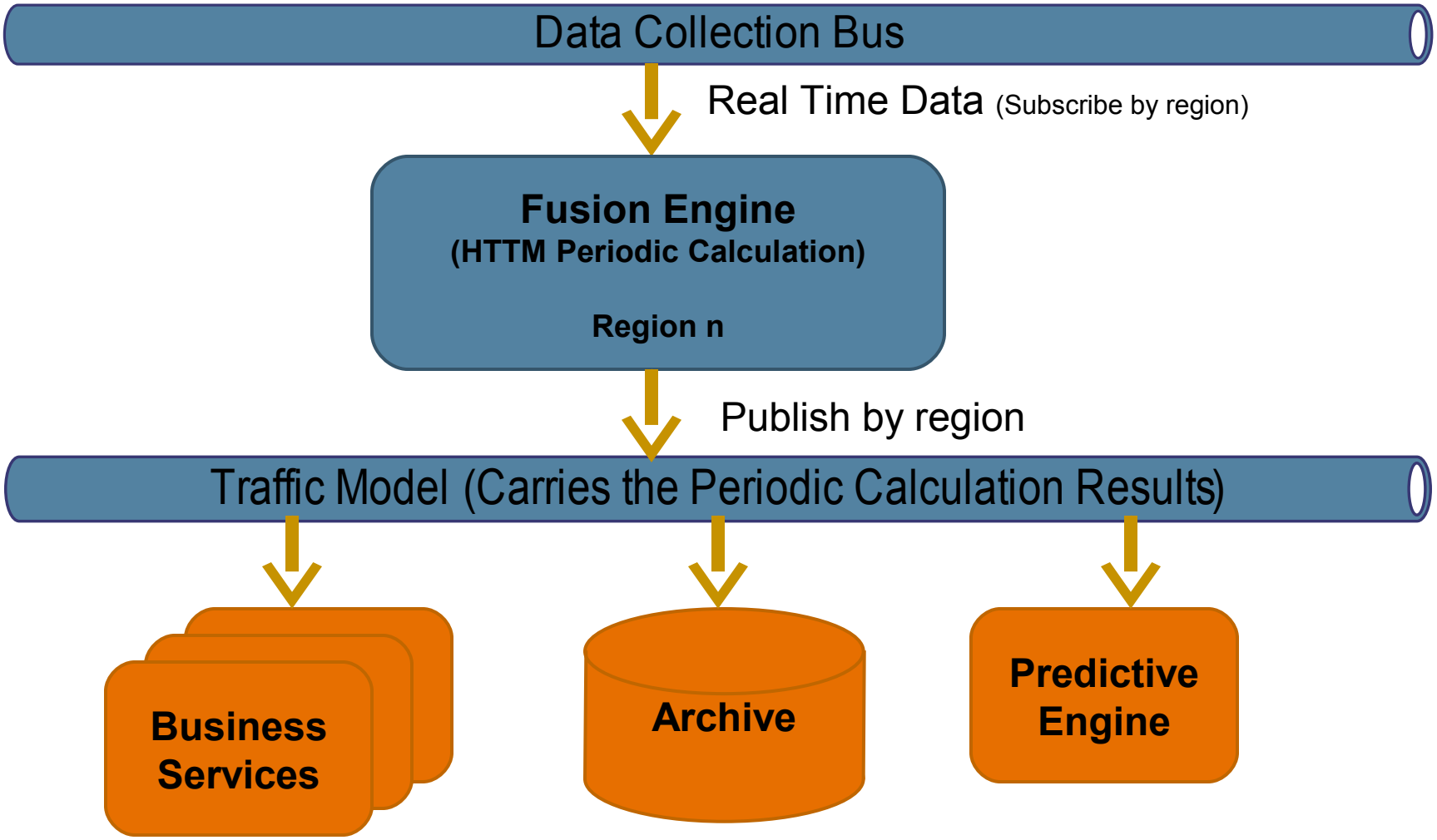
Do we calculate from scratch for each client request?

Is there a way to reuse previous calculations?

Data Fusion/Middleware

- Computationally efficient
  - Divide algorithm between:
    - Periodic calculations
    - Per request calculations
  - Make the periodic calculations do the bulk of the work.
  - Make the cycle of periodic calculations tunable yet short
  - Minimize the work done per request
- Immediate
  - From the time new data enters the system to the time it's available for end products is a maximum of the cycle time

Data Fusion/Middleware



Network bandwidth considerations

- Traffic Model – carries all of the periodic calculation results for the pertinent region
- ~1 Megabyte per region
- 1 Minute cycle time
- 100 regions

100 MBpm

Network bandwidth considerations

- On a 100 Mbps backplane we have approximately:
  10 bits/byte (when considering frame and protocol overhead) gives:

600 MBpm – theoretical Max

390 MBpm – practical limit

Theoretical Maximum - 6 clients

Practical Estimate - 4 clients

Network bandwidth considerations

- On a 1 Gbps backplane we have approximately:
  10 bits/byte (when considering frame and protocol overhead) gives:

6000 MBpm – theoretical Max

3900 MBpm – practical limit

Theoretical Maximum - 60 receivers

Practical Estimate - 39 receivers

Java Message Service (JMS) API – TCP with Centralized server

- Assume 1Gig bps – theoretical limit

$$\text{Subscribers} + 1 < 60$$

- Practically

$$\text{Subscribers} + 1 < 39$$

- Even 10 message destinations uses a significant percentage of available backplane bandwidth
- What if we have more subscribers?

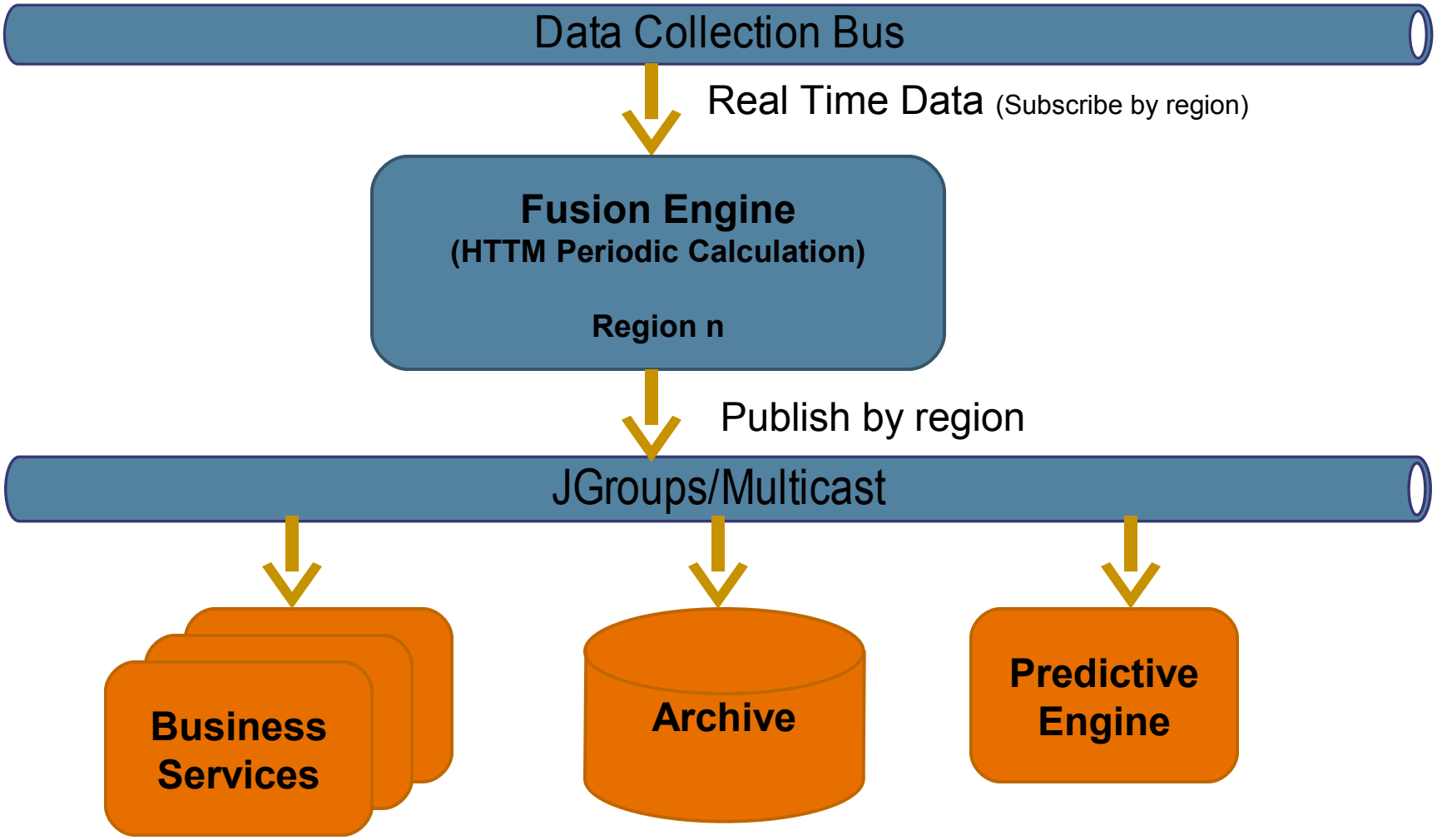
JGroups – Reliable Multicast

- Solution: Multicast!
  - Bandwidth doesn't increase with subscribers
  - Fixed message length per cycle
  - Bandwidth use increases linearly with cycle time
  - More flexibility when selecting cycle time

  - This keeps us at:

100MBpm

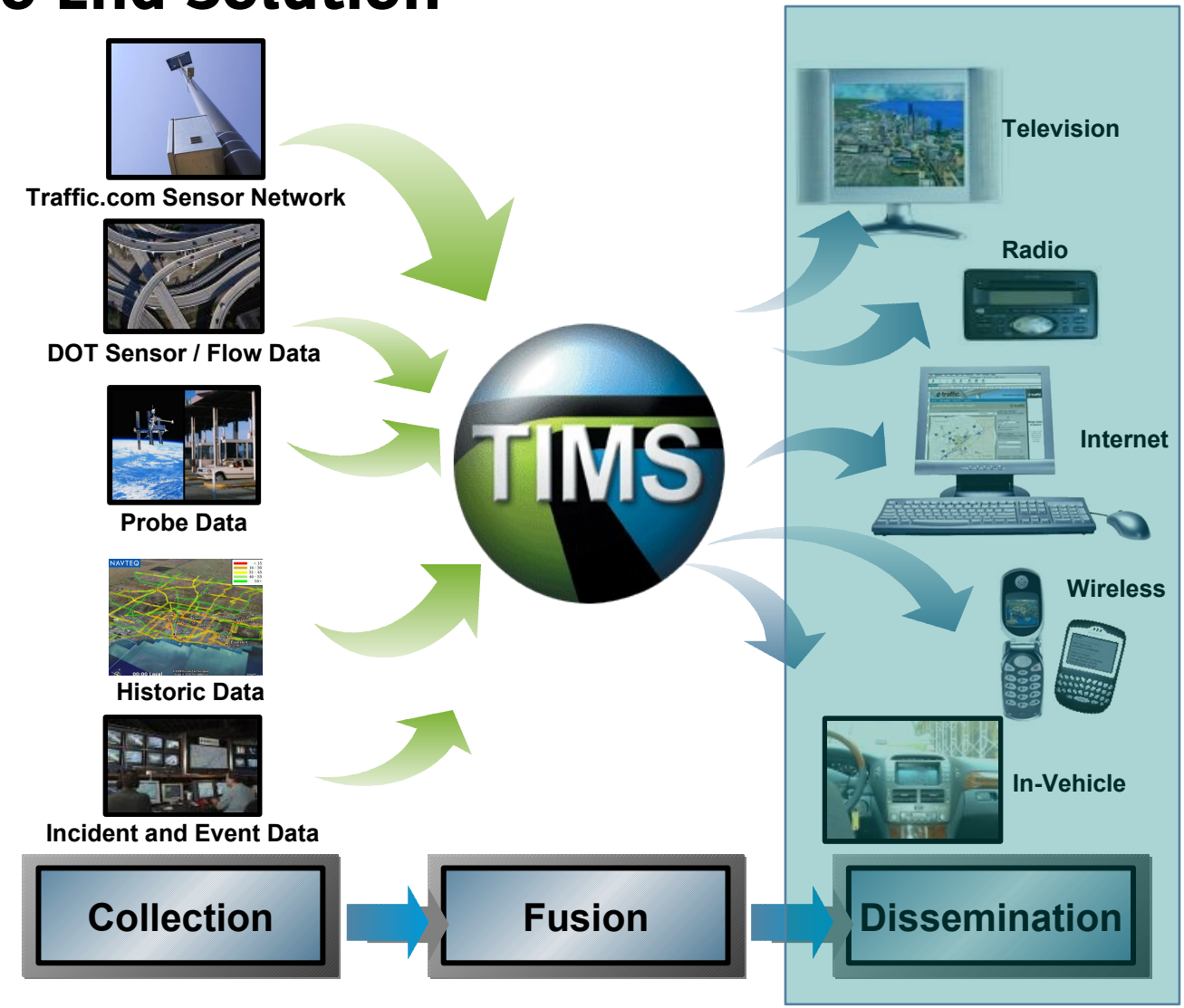
Data Fusion/Middleware



Agenda

- Introduction
- Data Collection
- Data Fusion/Middleware
- **Data Dissemination**
- Examples of Integrating Traffic Feeds
- Questions

End-to-End Solution

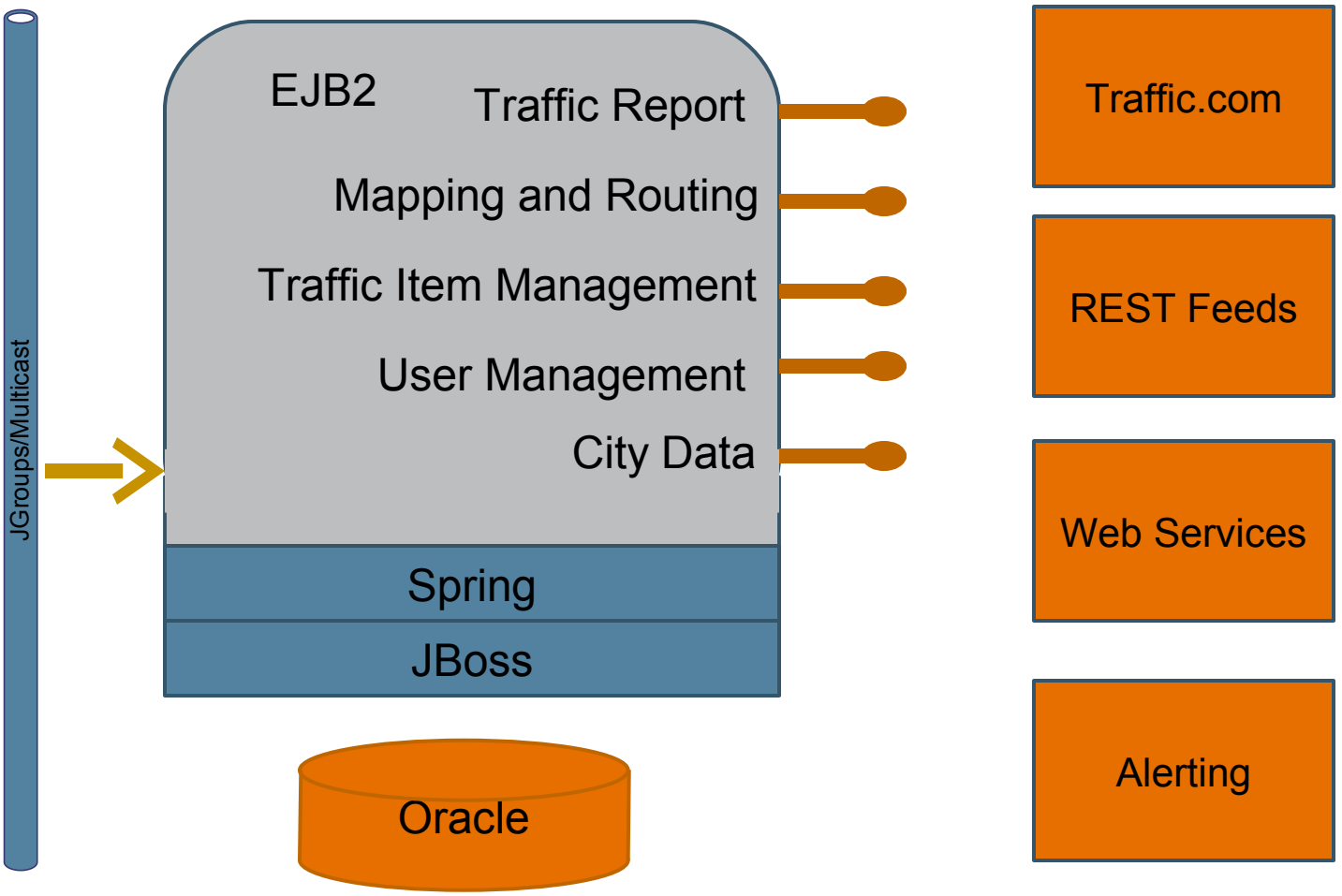


Visit: www.NN4D.com

Data Dissemination

- Data Feeds (examples are MSN Direct/Microsoft Live, Garmin, XM NavTraffic, Sirius Traffic)
  - WebServices
  - RESTful XML Services
  - Graphical integration
- Broadcast integration (Radio, NeXgen demonstration)
- One-to-one integration (Traffic.com, TrafficONE)
- Mobile integration (voice application, mobi.traffic.com, Sprint Java Platform, Micro Edition (J2ME™platform)/JamCast)
- Alerting infrastructure (three hundred thousand alerts sent daily to email, SMS, or voice calls)

Business Services



Business Services

- Client Server Communication Based on Enterprise JavaBeans™ (EJB™2) software
- Service Implementations are Spring Beans
- Services are Stateless (Stateless Session Beans)
  - Uncomplicated
  - Scalable linearly

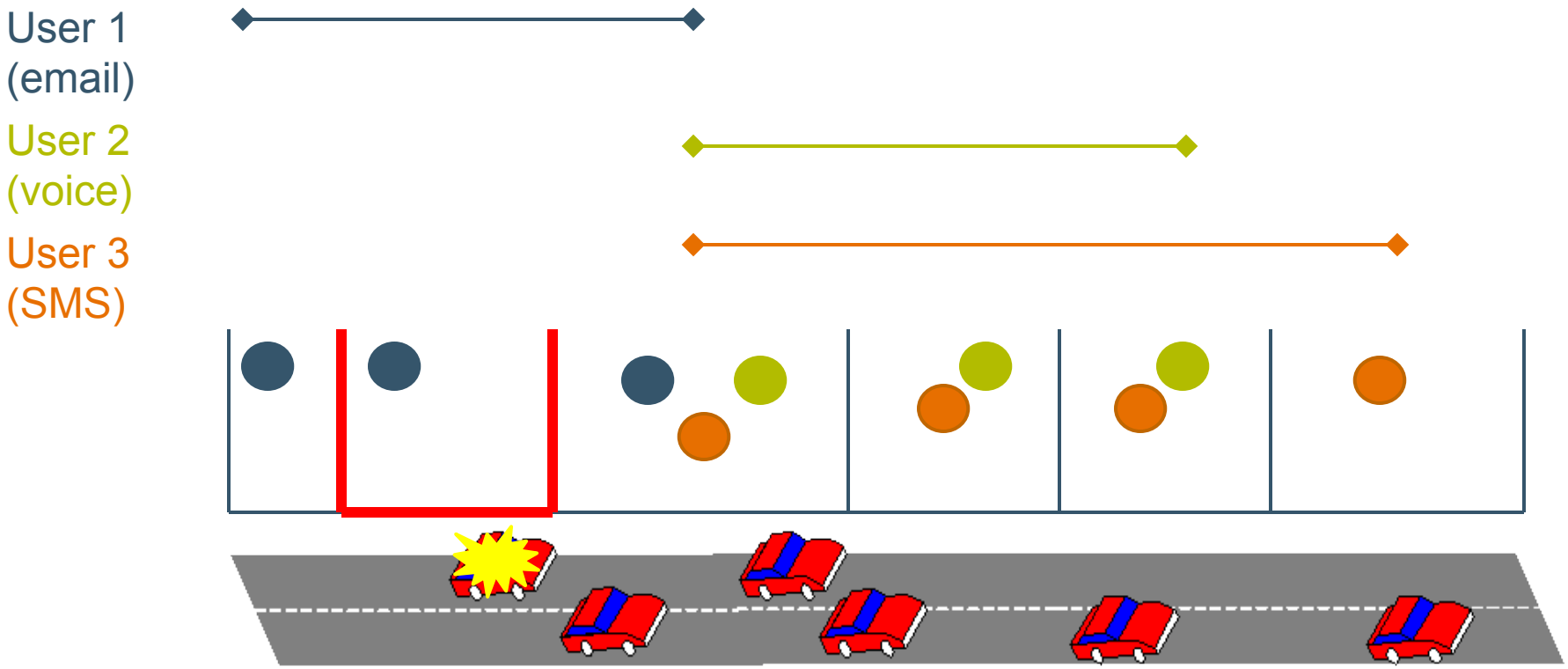
- Question: What's a stateless system?
  - A "Stateless" system simply puts more pressure on the DB

- Later versions of the system introduced Java Persistence API (JPA) with second level caching for state management

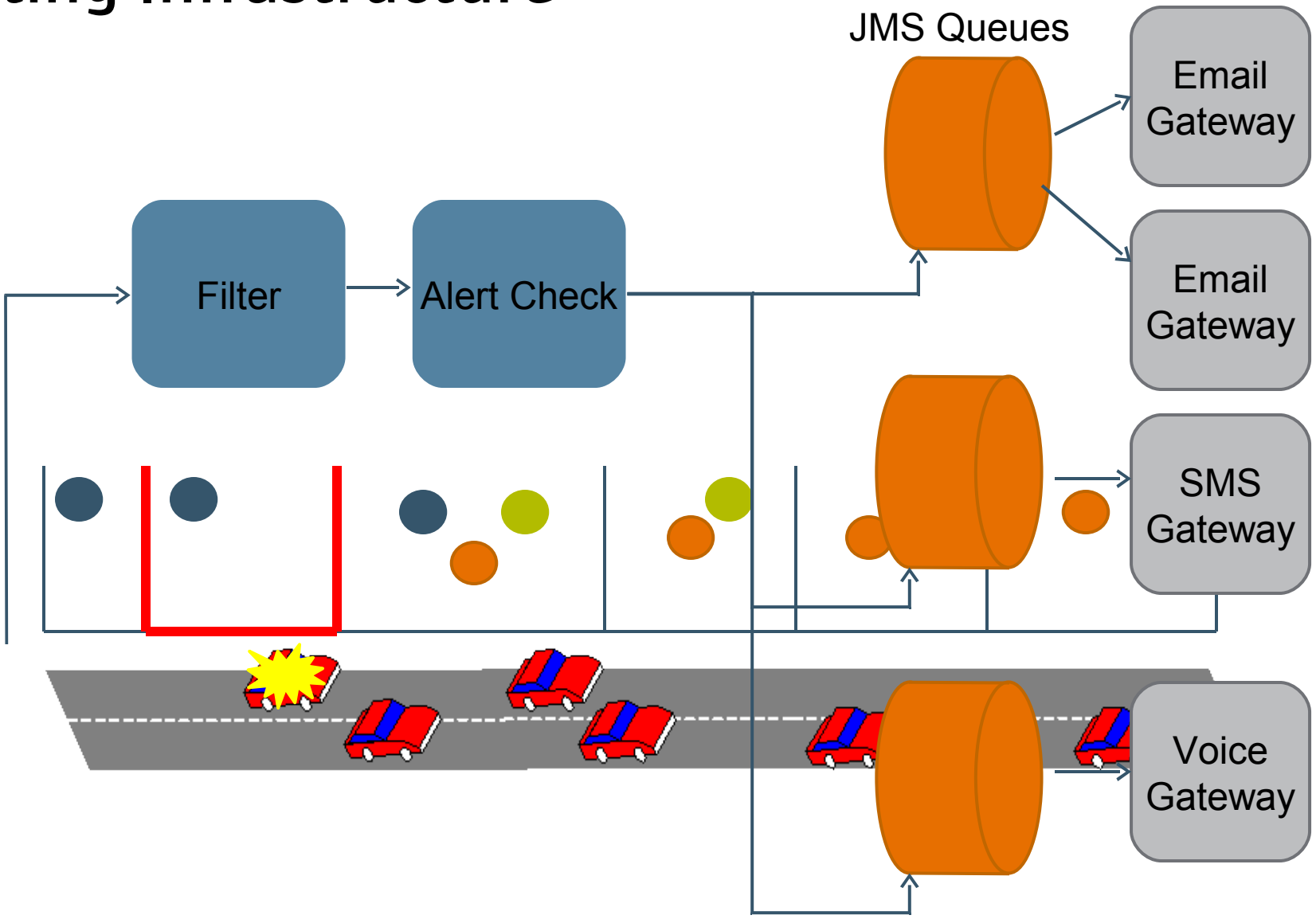
Alerting

- How to determine if totally customized routes have reached personalized thresholds on various traffic metrics
  - Travel time/delay
  - Incidents or criticality of incidents
- Impractical (doesn't scale) to consider every route periodically
- Need a data driven solution

Alerting Infrastructure



Alerting Infrastructure



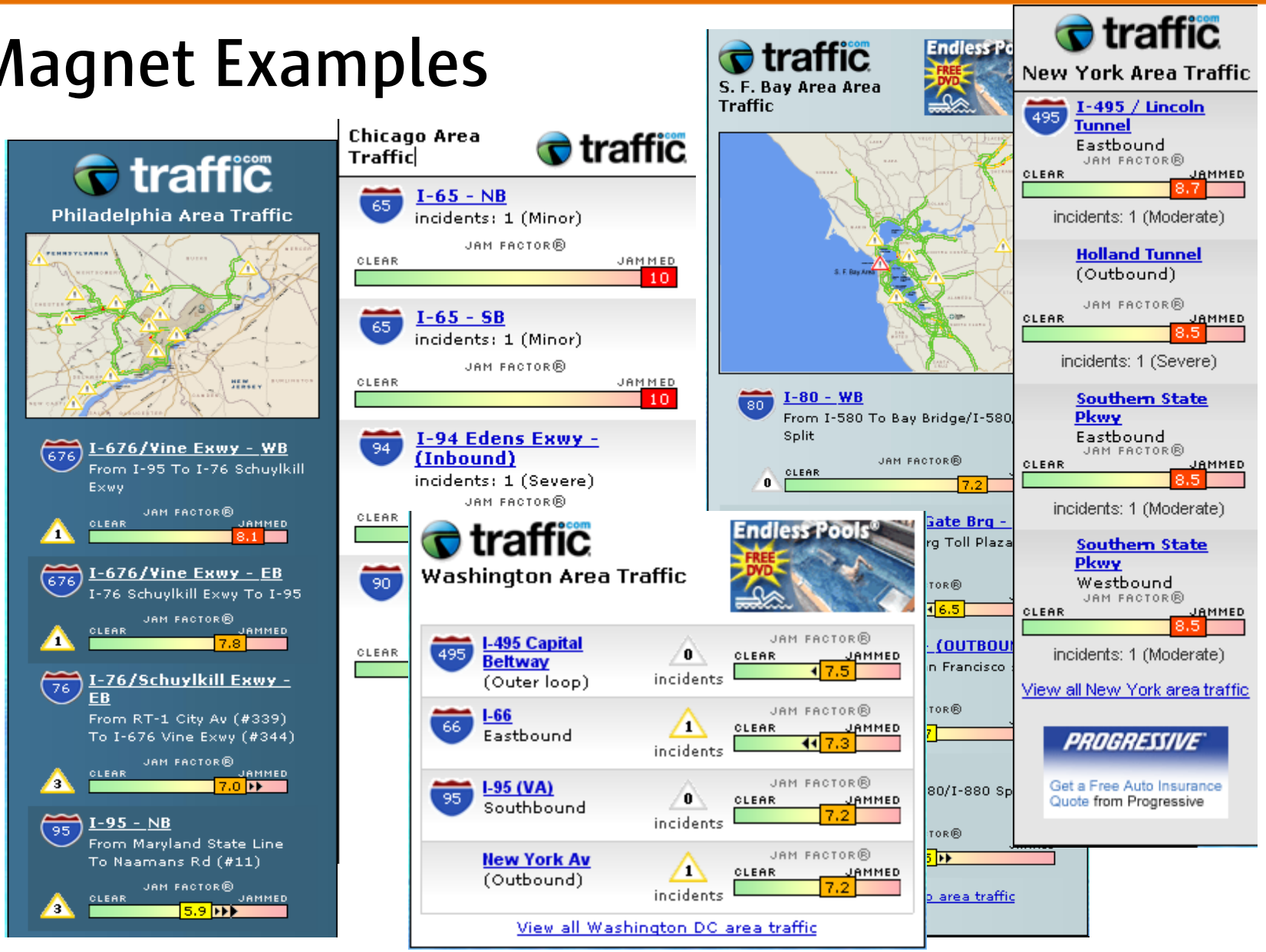
Agenda

- Introduction
- Data Collection
- Data Fusion/Middleware
- Data Dissemination
- **Examples of Integrating our Traffic Feeds**
- Questions

Magnets

- Custom traffic widgets for your website
- Easy to register, create and integrate into your website
- Thousands of magnets on 3rd party websites to date
- Can be integrated with our partner traffic sites (Traffic One)
- Go to: http://magnets.traffic.com

Magnet Examples



WebServices API

- Programmatic interface for traffic information
- Supports personalized information
- Features include:
  - User Profile (login, etc)
  - Incidents
  - Travel Time information
  - Alerting
  - Traffic Reports along a Drive
- Integrates with current website and can be used in conjunction with TrafficOne
- Availability currently requires agreement

Example – Web Service (Axis 1)

```
// Create the service locator - AXIS generated.
AuthenticationWebServiceLocator authLocator =
       new AuthenticationWebServiceLocator();

// Get the service stub from the locator.
AuthenticationStub authBinding =
       (AuthenticationStub)authLocator.getAuthentication();

// Set the request data object to contain the parameters.
AuthenticateWebServiceClient request =
       new AuthenticateWebServiceClient ();

request.setUserName (userName);
request.setPassword (password);
request.setDomain (domain);

// Make the call and get back the authentication context
WebServiceClientContext wscc =
      authBinding.authenticateWebServiceClient(request);
```

Visit: www.NN4D.com

Example – Web Service (Axis 1)

```
// Create the service locator for the metro service.
MetroWebServiceLocator metroLocator =
    new MetroWebServiceLocator();

// Get the metro service stub from the
MetroStub metroStub = (MetroStub)metroLocator.getMetro();

// set the client authentication context in the header
metroStub.setHeader(new
SOAPHeaderElement(wsdlUrl,"WebServiceClientContext", wscc));

// call the metro service to get the complete list of
//  metropolitan areas.
Metro[] metros = metroStub.getAllMetros();
```

Visit: www.NN4D.com

Example – Web Service (Axis 1)

```
// Create the service locator for the commute service.
CommuteWebServiceLocator commuteLocator =
                                new CommuteWebServiceLocator();

// Get the commute service stub from the
CommuteStub commuteStub = (CommuteStub)commuteLocator.getCommute();

// set the client authentication context in the header
commuteStub.setHeader(new SOAPHeaderElement(wsdlUrl,
                                "WebServiceClientContext", wscc));

// call the commute service to get the complete list of Key Routes
GetKeyRoutesByMetroRequest krRequestData =
                    new GetKeyRoutesByMetroRequest();

krRequest.setMetroId(selectedMetro.getMetroId());

KeyRoute[] keyRoutes = commuteStub.getKeyRoutesByMetro(krRequest);
```

Visit: www.NN4D.com

Example – Web Service (Axis 1)

```
// get the traffic report service set up
TrafficReportWebServiceLocator trafficReportLocator =
                        new TrafficReportWebServiceLocator();
TrafficReportStub trafficReportStub =

(TrafficReportStub)trafficReportLocator.getTrafficReport();
trafficReportStub.setHeader(new SOAPHeaderElement(wsdlUrl,
                                    "WebServiceClientContext", wscc));

for (KeyRoute keyRoute : keyRoutes)
{
      GetCommuteTrafficReportRequest trRequestData =
                  new GetCommuteTrafficReportRequest();
      trRequestData.setCommute(keyRoute.getCommute());
      CommuteTrafficReport tr =
                  trafficReportStub.getCommuteTrafficReport(trRequestData);

      System.out.println("Key Route:" + keyRoute.getKeyRouteDesc());
      System.out.println("   Avg Speed:" +
                  tr.getDigitalDetails().getAverageSpeed());
      System.out.println("   Number of Incidents:" +
                  tr.getTrafficItemCount());
}
```

Visit: www.NN4D.com

Web Service Features

Other traffic report attributes

- Jam Factor
- Jam Factor Trend
- Travel Time
- Delay
- Average Speed
- Confidence
- Traffic Items
  - Type
  - Criticality
  - Description

Other Services

- Commute Creation
- User Management
- City Meta Data
- Routing
- Mapping
- Alerting

Ajax/JavaScript™ Object Notation (JSON) technology (coming soon)

```
<html>
<head>
<script src="tajax.js"></script>
</head>
<body>
<script>

// create the object that will load our JSON
var jsonMetroLoader = new
                Nvt.Data.JSONLoader("metros");

// load the JSON
jsonMetroLoader.load(displayMetros);
```

Ajax/JSON (coming soon)

```
function displayMetro()
{
    if( !jsonMetroLoader.error )
    {
        var metros =
   jsonMetroLoader.getJSON().presentationObjects;

        metros.each(function(item) {
            /* display metro Info */
        })
    }
    else
        alert(jsonLoader.error);
}
```

Ajax/JSON (coming soon)

```
<html>
<head>
<script src="tajax.js"></script>
</head>
<body>
<script>

// create the object that will load our JSON
var jsonRoadLoader = new Nvt.Data.JSONLoader("roads");

// set required metro parameter
jsonRoadLoader.setMetroId(/* philadelphia */ 2);

// set filter and sorting attributes
jsonRoadLoader.setSortBy("driveReport.jamFactor.jamFactor");
jsonRoadLoader.setSortOrder("desc");
jsonRoadLoader.setMax(20);

// load the JSON
jsonRoadLoader.load(displayRoads);
```

Ajax/JSON (coming soon)

```
function displayRoads()
{
    if( ! jsonRoadLoader.error )
    {
        var roads =
   jsonRoadLoader.getJSON().presentationObjects;

        roads.each(function(item) {
            /* display road Info */
        })
    }
    else
        alert(jsonRoadLoader.error);
}
```

Ajax/JSON (coming soon)

- Road Information
  - Route number
  - Direction
- Traffic Information
  - Jam Factor
  - Delay
  - Average Speed
  - Traffic Items
    - Type
    - Criticality
    - Description

Also In the Works

XML REST Based API

Agenda

- Introduction
- Data Collection
- Data Fusion/Middleware
- Data Dissemination
- Examples of Integrating our Traffic Feeds
- Questions

Questions

- Brian Smyth, SVP Software Development
  - brian.smyth@navteq.com
- Jim Carroll, Chief Software Architect
  - jim.carroll@navteq.com

THANK YOU

Brian Smyth, SVP Software Development
Jim Carroll, Chief Software Architect

TS-6028

Visit: www.NN4D.com