



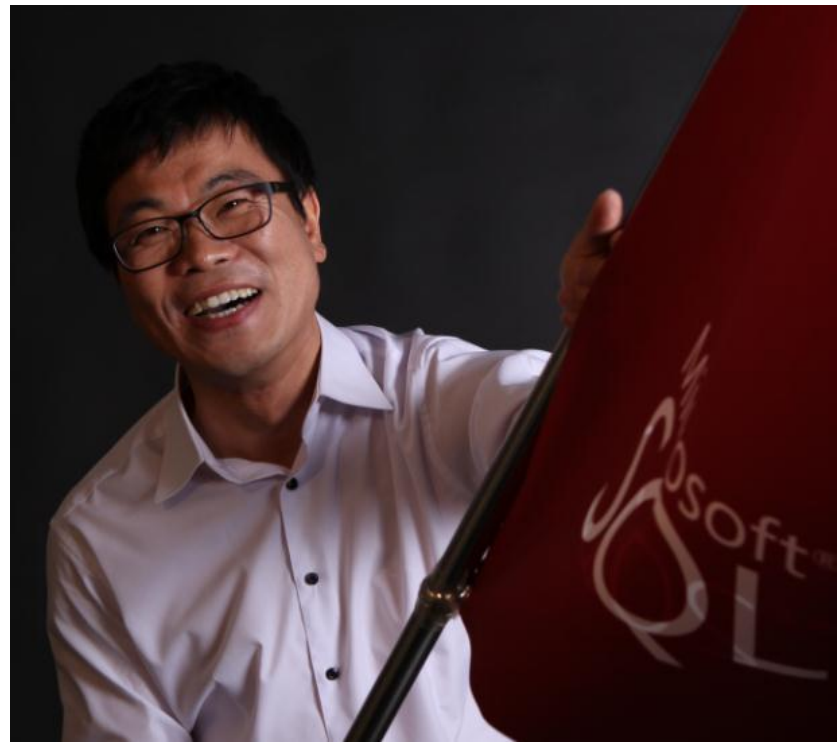
# StreamInsight를 통한 실시간 이벤트 데이터 처리

홍세환  
(주)인브레인

# 발표자 소개

## 홍세환

- (주) 인브레인
  - <http://www.inbrein.com>
  - 컨설팅사업부 부장
  - 개발 PM
  - C#



# 목차

- StreamInsight 소개
  - Complex Event Processing
  - StreamInsight Architecture
- Application Component
  - Event (Flow, Control and Definition)
  - LINQ & Window
  - Adapters
  - Development Model
  - Deployment Model
  - Monitoring & Event Flow Debugger
- Usage Example

## Event Driven Architecture (Event Processing Styles) – Wikipedia

- Simple event processing
  - Simple events can be created by a sensor detecting changes in tire pressures or ambient temperature.
- Event stream processing
  - Stream event processing is commonly used to drive the real-time flow of information in and around the enterprise, which enables in-time decision making.
- **Complex event processing**
  - CEP is commonly used to detect and respond to business anomalies, threats, and opportunities.

## Complex Event Processing

- Complex Event Processing (CEP) is the **continuous** and incremental processing of event streams from **multiple sources** based on declarative query and pattern specifications with **near-zero latency**

## The Goals of CEP

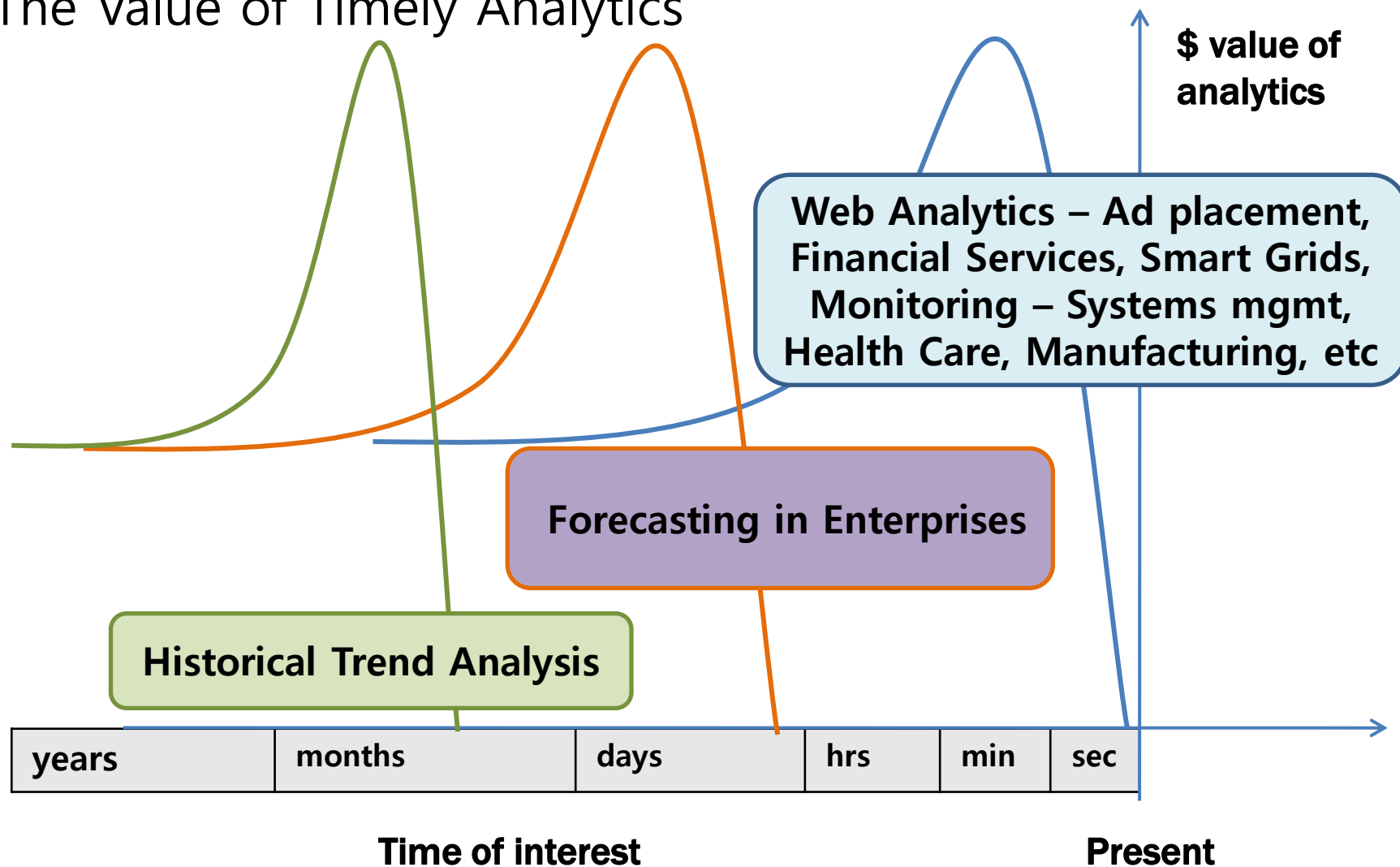
- Identify and detect from seemingly unrelated multiple events:
  - Meaningful patterns, Relationships
  - Trends, Gaps (expected events that did not occur)
  - Abstractions
  - Exceptions, Opportunities
- Analyze data without storing it first
- Trigger immediate response actions
- Mine events for new business KPIs

# StreamInsight 소개

(Complex Event Processing)



## The Value of Timely Analytics



# StreamInsight 소개

(Complex Event Processing)



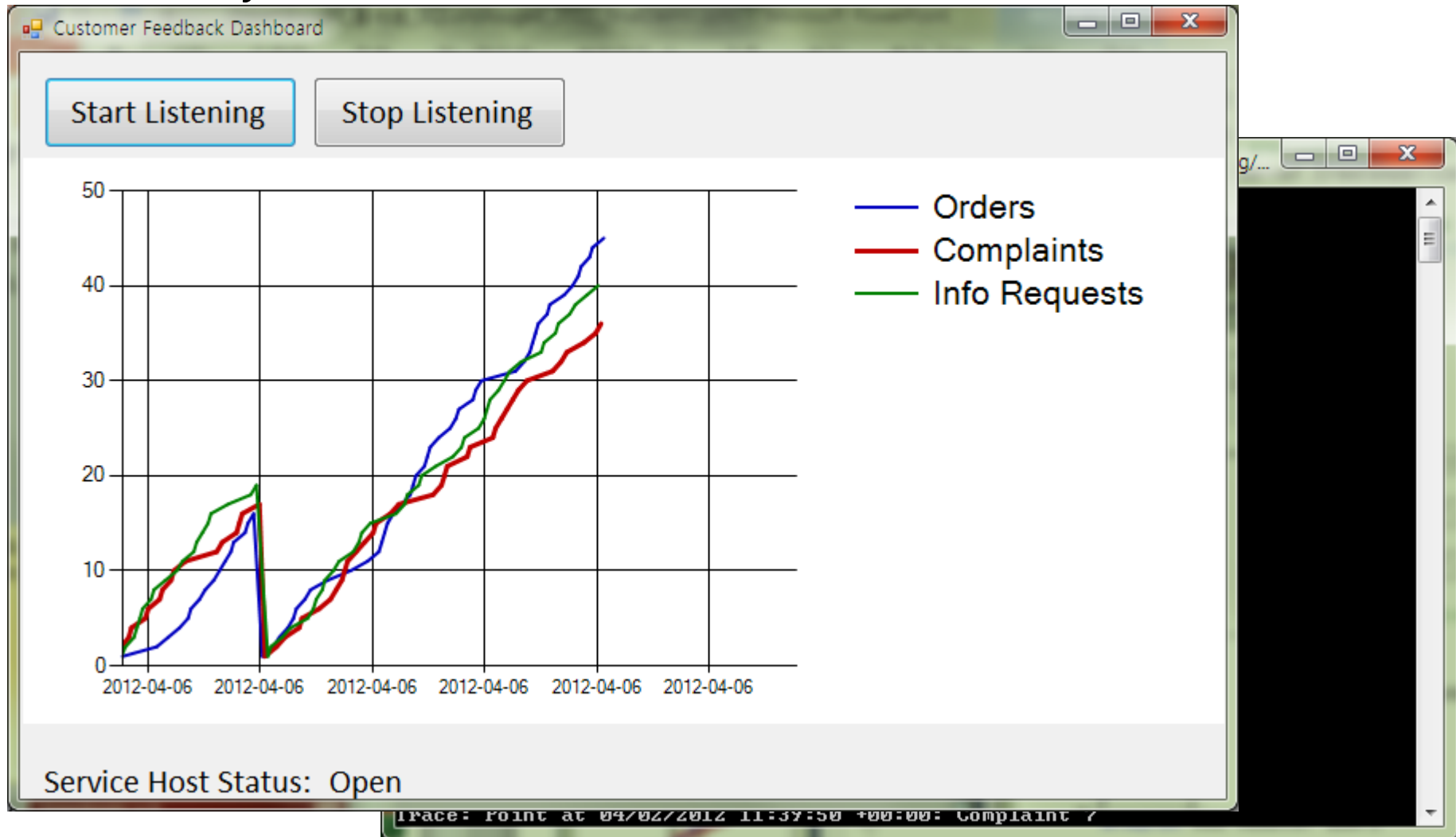
## Web Analytics Demo (Package\_StreamInsightSolution)

```
file:///C:/PlayFun/StreamInsight/Package_StreamInsightSolution/StreamInsightQuery/bin/Debug/...  
*** Create Server ***  
Trace: Point at 04/02/2012 11:39:34 +00:00: Complaint 1  
Trace: Point at 04/02/2012 11:39:34 +00:00: Order 2  
Trace: Point at 04/02/2012 11:39:36 +00:00: Complaint 2  
Trace: Point at 04/02/2012 11:39:36 +00:00: Order 4  
Trace: Point at 04/02/2012 11:39:36 +00:00: Info Request 1  
Trace: Point at 04/02/2012 11:39:38 +00:00: Complaint 2  
Trace: Point at 04/02/2012 11:39:38 +00:00: Info Request 3  
Trace: Point at 04/02/2012 11:39:38 +00:00: Order 6  
Trace: Point at 04/02/2012 11:39:40 +00:00: Complaint 4  
Trace: Point at 04/02/2012 11:39:40 +00:00: Order 6  
Trace: Point at 04/02/2012 11:39:40 +00:00: Info Request 5  
Trace: Point at 04/02/2012 11:39:42 +00:00: Complaint 5  
Trace: Point at 04/02/2012 11:39:42 +00:00: Info Request 7  
Trace: Point at 04/02/2012 11:39:42 +00:00: Order 7  
Trace: Point at 04/02/2012 11:39:44 +00:00: Complaint 5  
Trace: Point at 04/02/2012 11:39:44 +00:00: Info Request 10  
Trace: Point at 04/02/2012 11:39:44 +00:00: Order 5  
Trace: Point at 04/02/2012 11:39:46 +00:00: Complaint 4  
Trace: Point at 04/02/2012 11:39:46 +00:00: Order 5  
Trace: Point at 04/02/2012 11:39:46 +00:00: Info Request 11  
Trace: Point at 04/02/2012 11:39:48 +00:00: Complaint 6  
Trace: Point at 04/02/2012 11:39:48 +00:00: Info Request 9  
Trace: Point at 04/02/2012 11:39:48 +00:00: Order 5  
Trace: Point at 04/02/2012 11:39:50 +00:00: Complaint 7
```

# StreamInsight 소개

(Complex Event Processing)

## Web Analytics Demo (Package\_StreamInsightSolution)

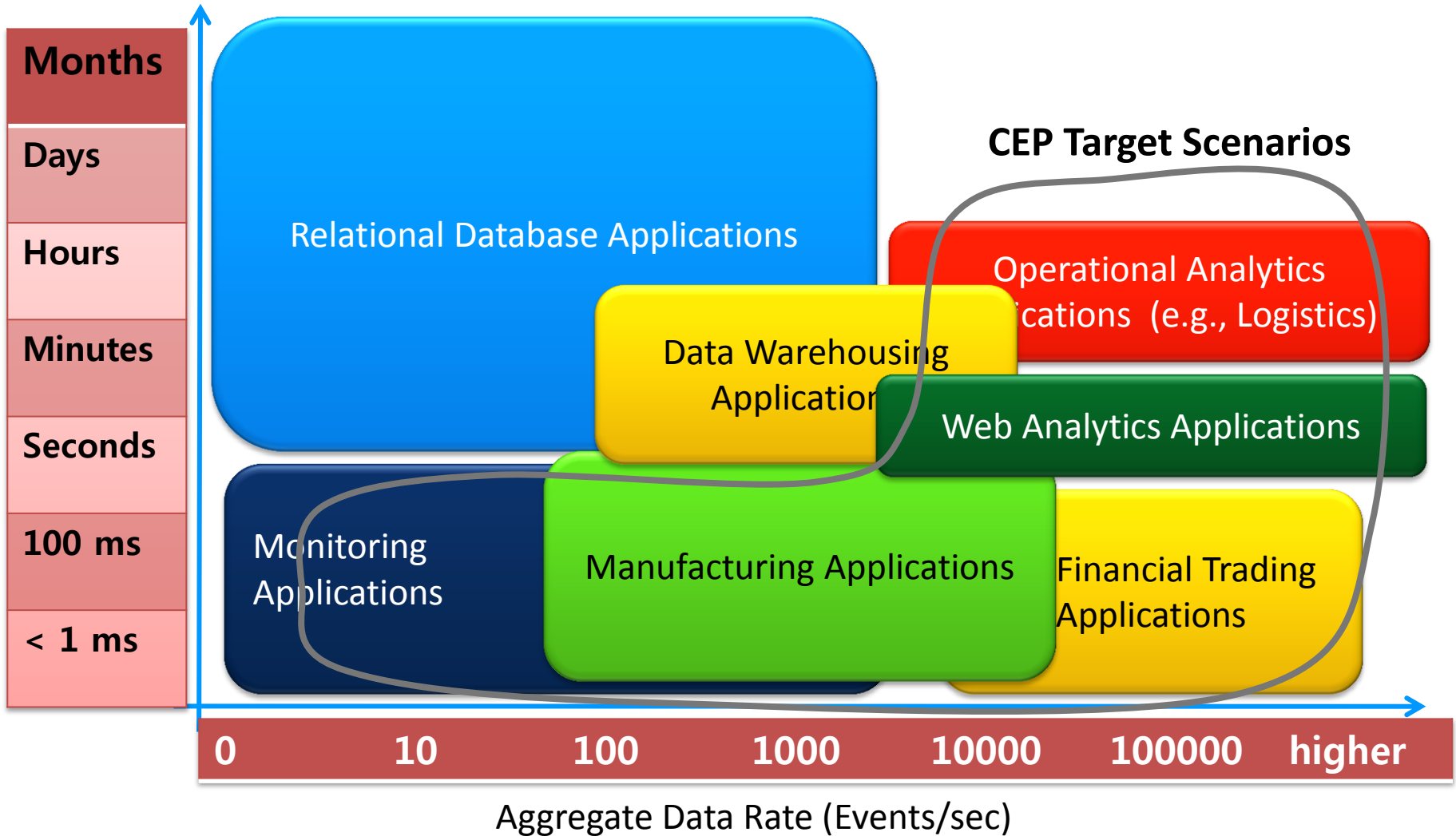




# StreamInsight 소개

(Complex Event Processing)

## CEP Sweet Spot



# StreamInsight 소개

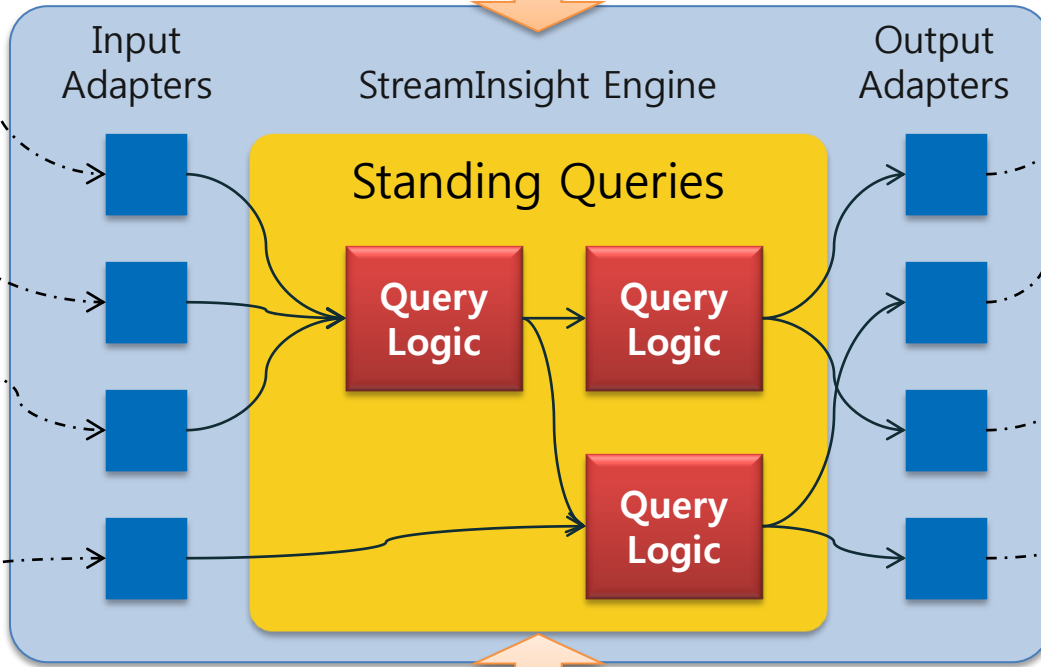
(StreamInsight Architecture)



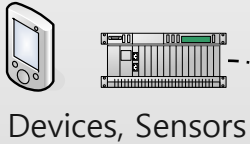
StreamInsight  
Application Development



## StreamInsight Application at Runtime



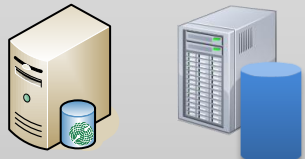
### Event sources



Devices, Sensors



Web servers



Event stores & Databases



Stock ticker, news feeds

### Event targets



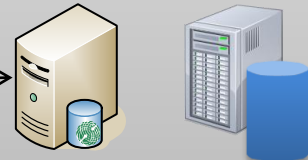
Pagers & Monitoring devices



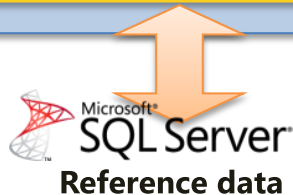
KPI Dashboards, SharePoint UI



Trading stations



Event stores & Databases



Reference data

# StreamInsight 소개

(StreamInsight Architecture)



## StreamInsight VS. DBMS

Analytical results need to reflect important changes in business reality **immediately** and enable responses to them with **minimal latency**

	Database	CEP
Queries	Ad hoc on stored data	Continuous standing queries
Latency	Seconds	Milliseconds
Data Rate	Hundreds per Second	Tens of thousands per second
Query Semantics	Declarative relational analytics	Declarative relational <i>and temporal</i> analytics



# StreamInsight 소개

(StreamInsight Architecture)



## SQL Server Capabilities by Edition

Workload	Standard	Enterprise	Datacenter	Parallel Data Warehouse
Custom/Packaged OLTP Apps	4 procs, 64GB RAM, Backup Compression	8 procs, 2TB RAM, Adv. Security, Backup Compression	>8 procs, OS Max, Adv. Security, Backup Compression	N/A
Server Consolidation	1 VM/license	4 VMs/license, Resource Governor App & Multi-Server Mgmt (up to 25 instances)	Unlimited Virtualization, Resource Governor, App & Multi-Server Mgmt (> 25 instances)	N/A
Data Warehousing		Scale-Up DW, Data Compression  10s of TBs, Up to 30 TB with FastTrack	Scale-Up DW, Data Compression  10s of TBs	Scale-Out DW  10s - 100s of TBs
Business Intelligence	Dept/Team BI	Enterprise-Scale BI, Master Data Services, PowerPivot Mgmt	Enterprise-Scale BI, Master Data Services, PowerPivot Mgmt	Integrated with SSIS, SSAS and SSRS
<b><u>StreamInsight</u></b>	<b><u>&lt; 5000 events/sec &amp; &gt; 5 sec latency</u></b>	<b><u>&lt; 5000 events/sec &amp; &gt; 5 s latency</u></b>	<b><u>&gt; 5000 events/sec &amp; &lt; 5 s latency</u></b>	<b><u>Future coverage</u></b>

# Application Component

- Event Stream Concept
- .NET 3.5 SP1, 4.0 and CLR for the runtime
- Visual Studio for developer productivity (IntelliSense)

- C# for apps development

- Adapter, Object Model
- Diagnostic APIs
- IEnumerable – Pull Model
- IObservable – Push Model



- LINQ for Query Language surface

- Anonymous Method
- Lambda Expression
- SQOs (Standard Query Operators)

- Event Flow Debugger



Microsoft®  
**StreamInsight**™  
Event Flow Debugger

# Application Component

## Event (Flow, Control and Definition)



### Streams

- All data organized into Streams, Potentially unending
- Data may change over time
- Often represent the same value over time

### Events

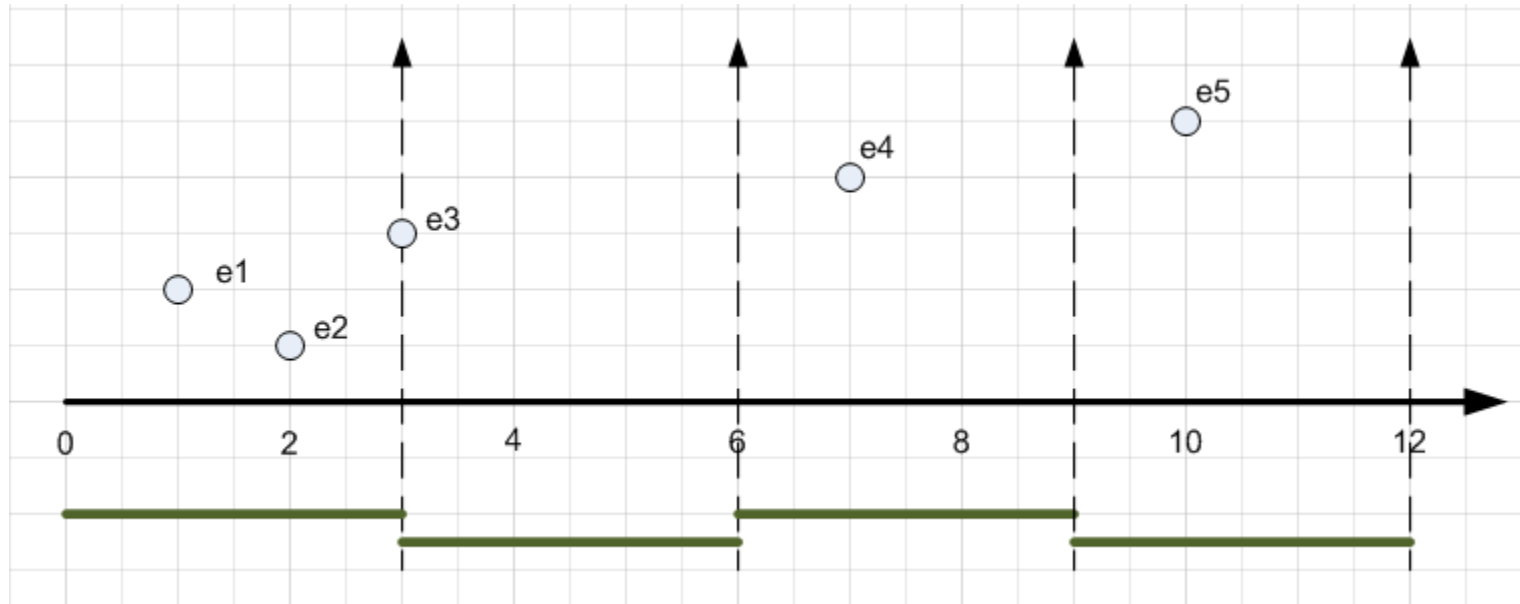
- Data in streams is packaged into events
- Two parts to an event
  - Header (kind of event and timestamps)
  - Payload (.NET data structure)
- Timestamps are DateTimeOffset data type
  - All times normalized to UTC in server

Timestamps/ MetadataC	Long pumpID	String Type	String Location	Double flow	Double pressure
...	...	...	...	...	...

# Application Component

## Event (Flow, Control and Definition)

### Event Model (Event Shapes) - Point Event

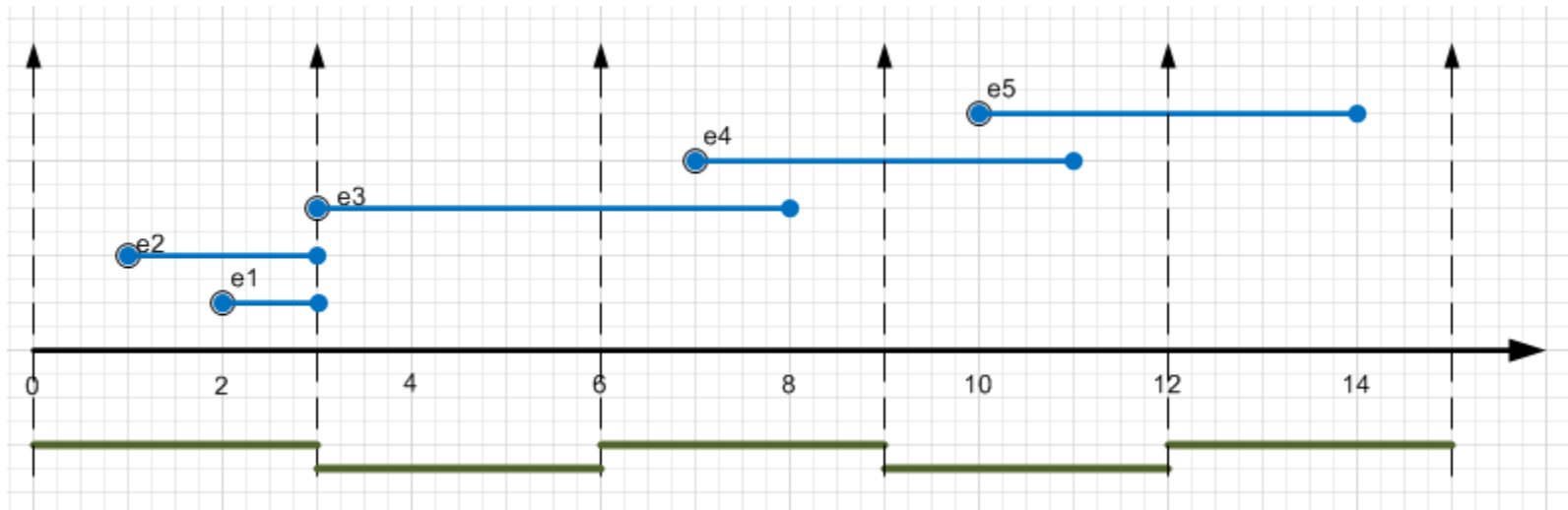


EventKind	StartTime	EndTime	Payload
INSERT	2009-12-27 02:04:00.213	2009-12-27 02:04:00.213 + c	EU-23423-12
INSERT	2009-12-27 02:04:04.329	2009-12-27 02:04:04.329 + c	EU-23423-15
INSERT	2009-12-27 02:04:04.234	2009-12-27 02:04:04.234 + c	EU-23423-18

# Application Component

## Event (Flow, Control and Definition)

### Event Model (Event Shapes) - Interval Event



EventKind	StartTime	EndTime	Payload
INSERT	2009-12-27 02:04:00.213	2009-12-27 02:04:04.329	EU-23423-12
INSERT	2009-12-27 02:04:04.329	2009-12-27 02:04:08.234	EU-23423-15
INSERT	2009-12-27 02:04:04.234	2009-12-27 02:04:04.523	EU-23423-18

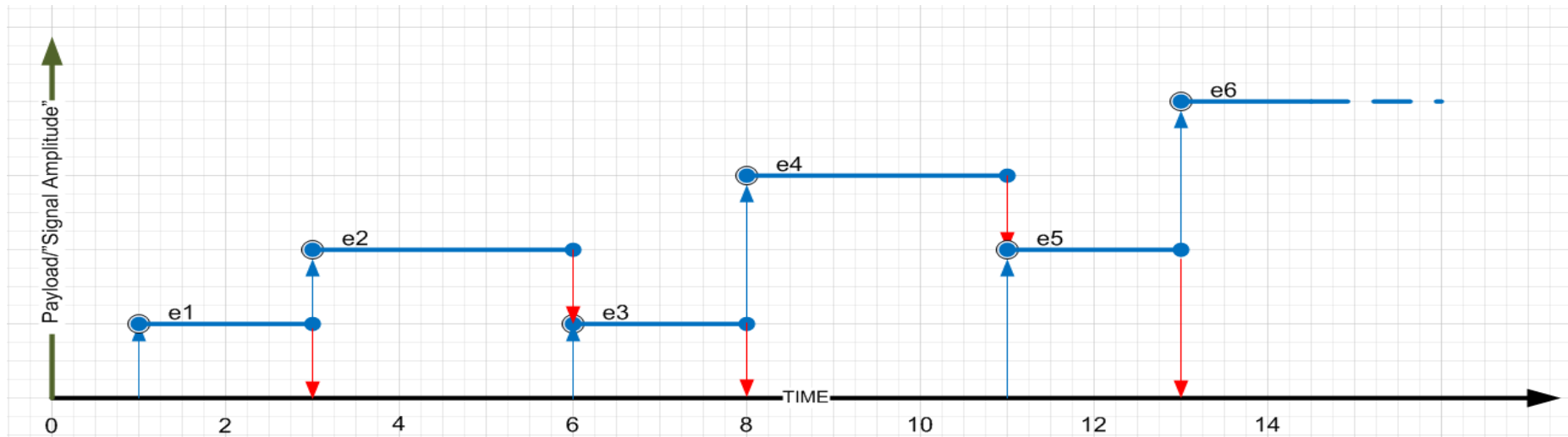


# Application Component

## Event (Flow, Control and Definition)



### Event Model (Event Shapes) - Edge Event



EventKind	EdgeType	StartTime	EndTime	Payload
INSERT	Start	2009-12-27 02:04:00.213	$\infty$	EU-23423-12
INSERT	End	2009-12-27 02:04:00.213	2009-12-27 02:04:04.329	EU-23423-12
INSERT	Start	2009-12-27 02:04:04.234	$\infty$	EU-23423-18
INSERT	End	2009-12-27 02:04:04.234	2009-12-27 02:04:08.238	EU-23423-18

# Application Component

## Event (Flow, Control and Definition)



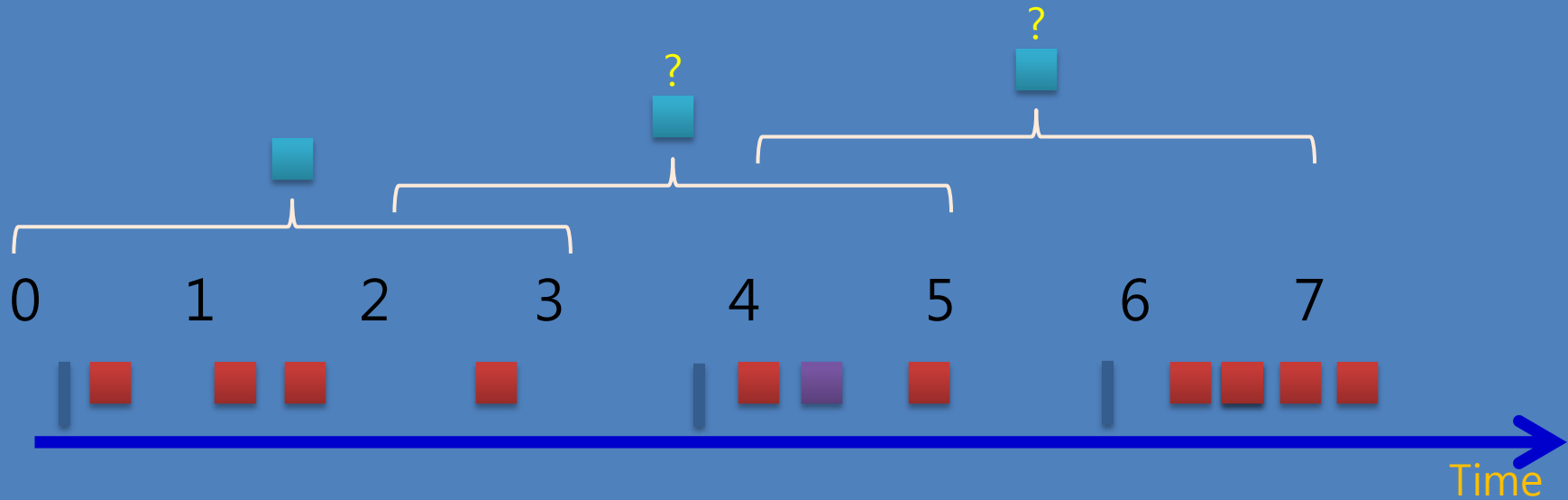
### Event Kind

- Two event kinds
  - INSERT (new data for the stream)
  - CTI (Current Time Increment)
- C T I
  - Added to input stream
    - Typically added by input adapter
    - Can be declaratively added in query binding
  - Processing out of order data
  - Responsive stream
- Update – in Edge Event but Insert EndTime
- Delete – no concept of deleting, ReleaseEvent

# Application Component

Event (Flow, Control and Definition)

## Stream Liveliness



# Application Component

## LINQ & Window



## LINQ Demo

(MSSQL2008R2DevelopersTrainingKit\Labs\SQL10R2UPD05-HOL-01\Source\Assets\DisplayQueryVersion)

Highway Monitor

Stop  Auto-Scroll  Ignore CTI

	Command	Timestamp	Lane	TagId	VehicleType
	INSERT	2012-04-06 오전 12:22:56 +00:00	6	1403864547	Truck
	INSERT	2012-04-06 오전 12:22:57 +00:00	5	705555557	Car
	INSERT	2012-04-06 오전 12:22:57 +00:00	1	916470080	Bus
	INSERT	2012-04-06 오전 12:22:57 +00:00	6	725958970	Truck
	INSERT	2012-04-06 오전 12:22:57 +00:00	5	1383091759	Car
	INSERT	2012-04-06 오전 12:22:58 +00:00	1	1381480201	Ambulance
	INSERT	2012-04-06 오전 12:22:58 +00:00	6	220677196	Taxi
	INSERT	2012-04-06 오전 12:22:58 +00:00	4	1652305411	Car
	INSERT	2012-04-06 오전 12:22:58 +00:00	0	2130734117	Car
	INSERT	2012-04-06 오전 12:22:59 +00:00	3	1429755923	Taxi
	INSERT	2012-04-06 오전 12:22:59 +00:00	7	347872102	Bus
	INSERT	2012-04-06 오전 12:22:59 +00:00	7	969806795	Car

// 5 - filtering

```
var queryOutput = from e in input
```

```
    //where e.VehicleTypeId = 2
```

```
    select new { e.Lane, e.TagId, VehicleType = TollPointEvent.VehicleTypeName(e.VehicleTypeId) };
```

# Application Component

## LINQ & Window



## LINQ Demo

(MSSQL2008R2DevelopersTrainingKit\Labs\SQL10R2UPD05-HOL-01\Source\Assets\DisplayQueryVersion)

Highway Monitor

Stop  Auto-Scroll  Ignore CTI

	Command	Timestamp	Lane	TagId	VehicleType
	INSERT	2012-04-06 오전 12:22:56 +00:00	6	1403864547	Truck
	INSERT	2012-04-06 오전 12:22:57 +00:00	5	705555557	Car
	INSERT	2012-04-06 오전 12:22:57 +00:00	1	916470080	Bus
	INSERT	2012-04-06 오전 12:22:57 +00:00	6	725958970	Truck
	INSERT	2012-04-06 오전 12:22:57 +00:00	5	1383091759	Car
	INSERT	2012-04-06 오전 12:22:58 +00:00	1	1381480201	Ambulance
	INSERT	2012-04-06 오전 12:22:58 +00:00	6	220677196	Taxi
	INSERT	2012-04-06 오전 12:22:58 +00:00	4	1652305411	Car
	INSERT	2012-04-06 오전 12:22:58 +00:00	0	2130734117	Car
	INSERT	2012-04-06 오전 12:22:59 +00:00	3	1429755923	Taxi
	INSERT	2012-04-06 오전 12:22:59 +00:00	7	347872102	Bus
	INSERT	2012-04-06 오전 12:22:59 +00:00	7	969806795	Car

// 5 - filtering

```
var queryOutput = from e in input
```

```
    where e.VehicleTypeId == 2
```

```
    select new { e.Lane, e.TagId, VehicleType = TolIPointEvent.VehicleTypeName(e.VehicleTypeId) };
```

# Application Component

## LINQ & Window



## LINQ Demo

(MSSQL2008R2DevelopersTrainingKit\Labs\SQL10R2UPD05-HOL-01\Source\Assets\DisplayQueryVersion)

Highway Monitor

Stop  Auto-Scroll  Ignore CTI

	Command	Timestamp	Lane	TagId	VehicleType
▶	INSERT	2012-04-06 오전 12:23:36 +00:00	3	1722712655	Truck
	INSERT	2012-04-06 오전 12:23:37 +00:00	1	1323228034	Truck
	INSERT	2012-04-06 오전 12:23:37 +00:00	3	1911441173	Truck
	INSERT	2012-04-06 오전 12:23:38 +00:00	7	1051601669	Truck
	INSERT	2012-04-06 오전 12:23:39 +00:00	6	1332905677	Truck
	INSERT	2012-04-06 오전 12:23:41 +00:00	7	1194600731	Truck
	INSERT	2012-04-06 오전 12:23:42 +00:00	4	677647227	Truck
*					

```
// 5 - filtering
```

```
var queryOutput = from e in input
```

```
    where e.VehicleTypeId == 2
```

```
    select new { e.Lane, e.TagId, VehicleType = TolIPointEvent.VehicleTypeName(e.VehicleTypeId) };
```

# Application Component

## LINQ & Window



### LINQ Query Example

LINQ Example – JOIN, PROJECT, FILTER:

```
from e1 in MyStream1  
join e2 in MyStream2  
on e1.ID equals e2.ID
```

```
where e1.f2 = "foo"
```

```
select new { e1.f1, e2.f4 };
```

Join

Filter

Project

LINQ Example – GROUP&APPLY, WINDOW:

```
from e3 in MyStream3  
group e3 by e3.i into SubStreams  
from s4 in SubStreams
```

```
from e4 in s4.HoppingWindow(FiveMinutes,ThreeSeconds)
```

```
select new {  
    pl = new MyNewPayload(e4.i, e4.f)};
```

Grouping

Window

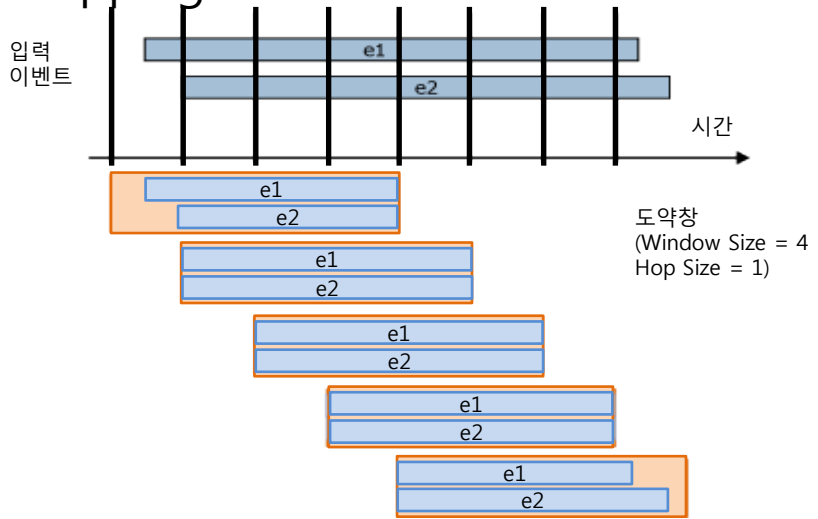
## CEP Query Features

- Operators over streams
  - PROJECT
  - JOIN
  - EXISTS
  - FILTER
  - GROUP & APPLY
  - SUM, COUNT, ...
  - TOP-K
  - **Temporal operations - window**
- Extensibility – to add new domain-specific operators
- Queries are written over specific event types
- Support for streaming data, reference data (lookup), and historical data (replay)

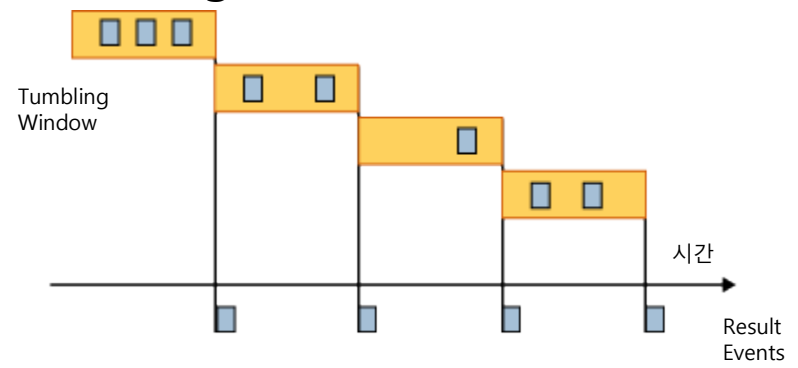


### Query Elements : Event Windows

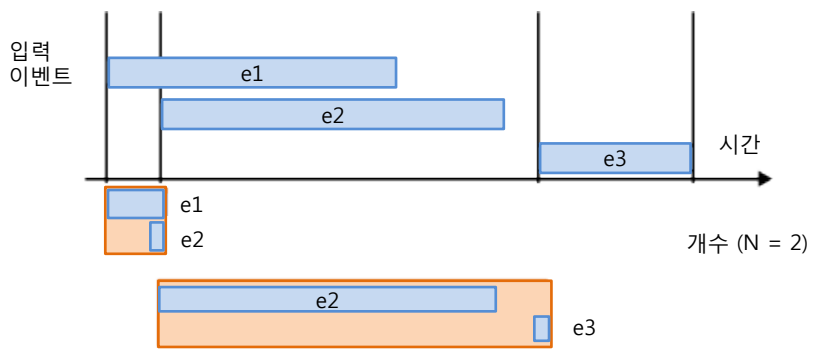
Hopping Window



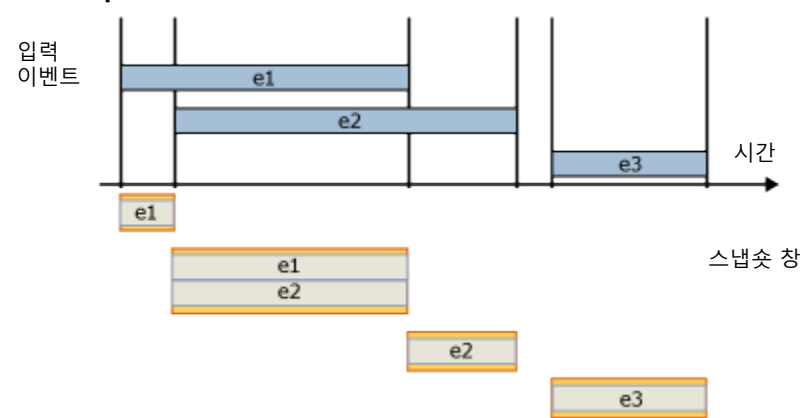
Tumbling Window



Count Window



Snapshot Window



## Query Elements : Event Window Demo

```
var vehicleSpeeds = from e in input
                    select new
                    {
                        e.DirectionId,
                        Speed = 60.0 * 60.0 / e.MillisecondsToPas
                    };
```

```
var topSpeeds = (from w in vehicleSpeeds.TumblingWindow
                (TimeSpan.FromSeconds(10))
                from e in w
                orderby e.Speed descending
                select e).Take(5, e => new
                {
                    e.Payload.DirectionId,
                    e.Payload.Speed,
                    e.Rank
                });
```

# Application Component

## LINQ & Window



## Query Elements : Event Window Demo

Highway Monitor

Stop  Auto-Scroll  Ignore CTI

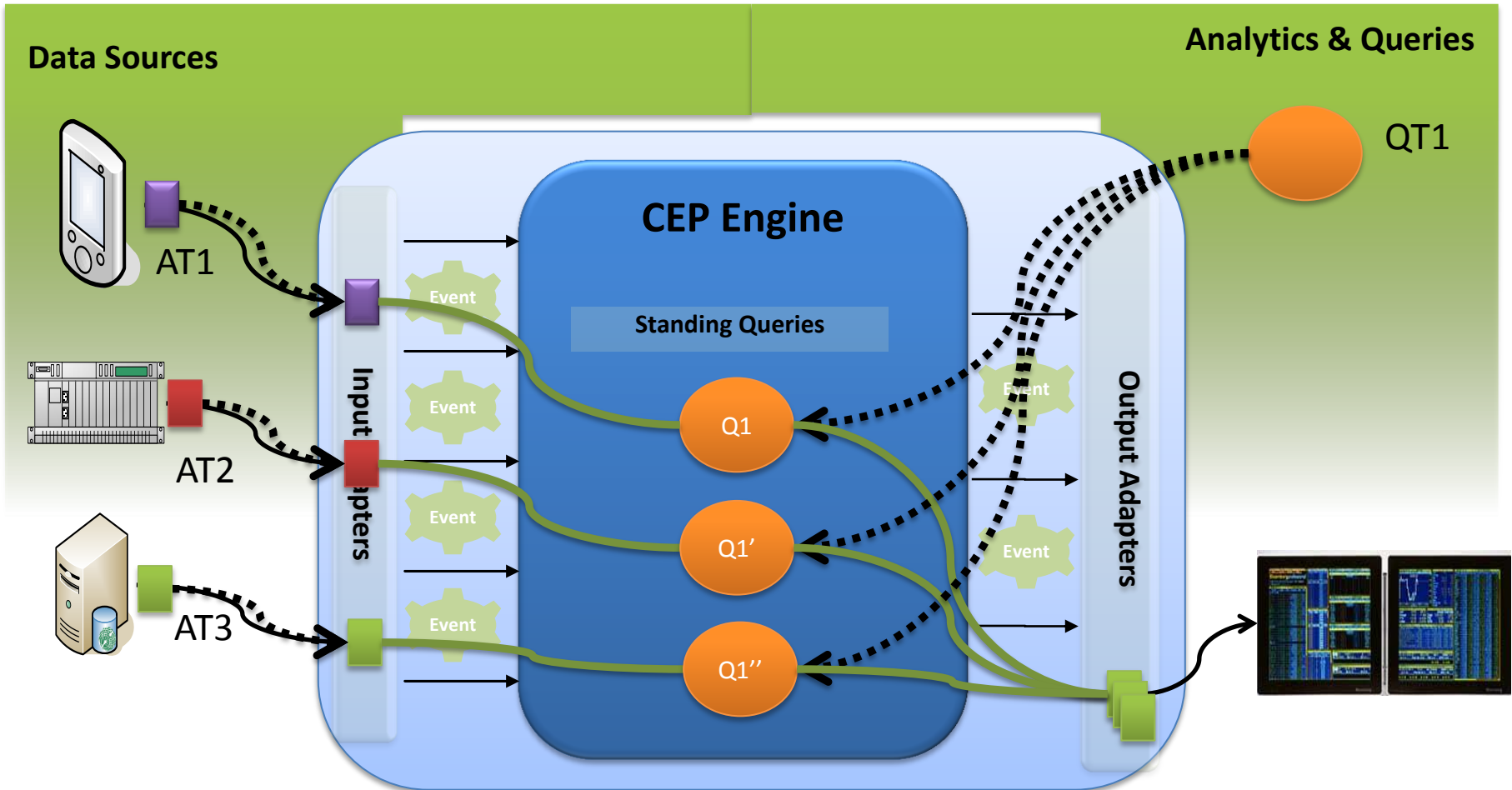
Command	Timestamp	DirectionId	Rank	Speed
INSERT	2012-04-06 오전 1:30:00 +00:00	0	1	109.0909090
INSERT	2012-04-06 오전 1:30:00 +00:00	0	2	105.8823529
INSERT	2012-04-06 오전 1:30:00 +00:00	0	5	83.7209302
INSERT	2012-04-06 오전 1:30:00 +00:00	1	3	87.8048780
INSERT	2012-04-06 오전 1:30:00 +00:00	1	4	85.7142857
INSERT	2012-04-06 오전 1:30:03 +00:00	0	1	109.0909090
INSERT	2012-04-06 오전 1:30:03 +00:00	0	3	100
INSERT	2012-04-06 오전 1:30:03 +00:00	0	5	80
INSERT	2012-04-06 오전 1:30:03 +00:00	1	1	109.0909090
INSERT	2012-04-06 오전 1:30:03 +00:00	1	4	90
INSERT	2012-04-06 오전 1:30:06 +00:00	0	1	109.0909090
INSERT	2012-04-06 오전 1:30:06 +00:00	0	2	105.8823529

```
e.Payload.DirectionId,  
e.Payload.Speed,  
e.Rank  
});
```

# Application Component

LINQ & Window

## Query Binding

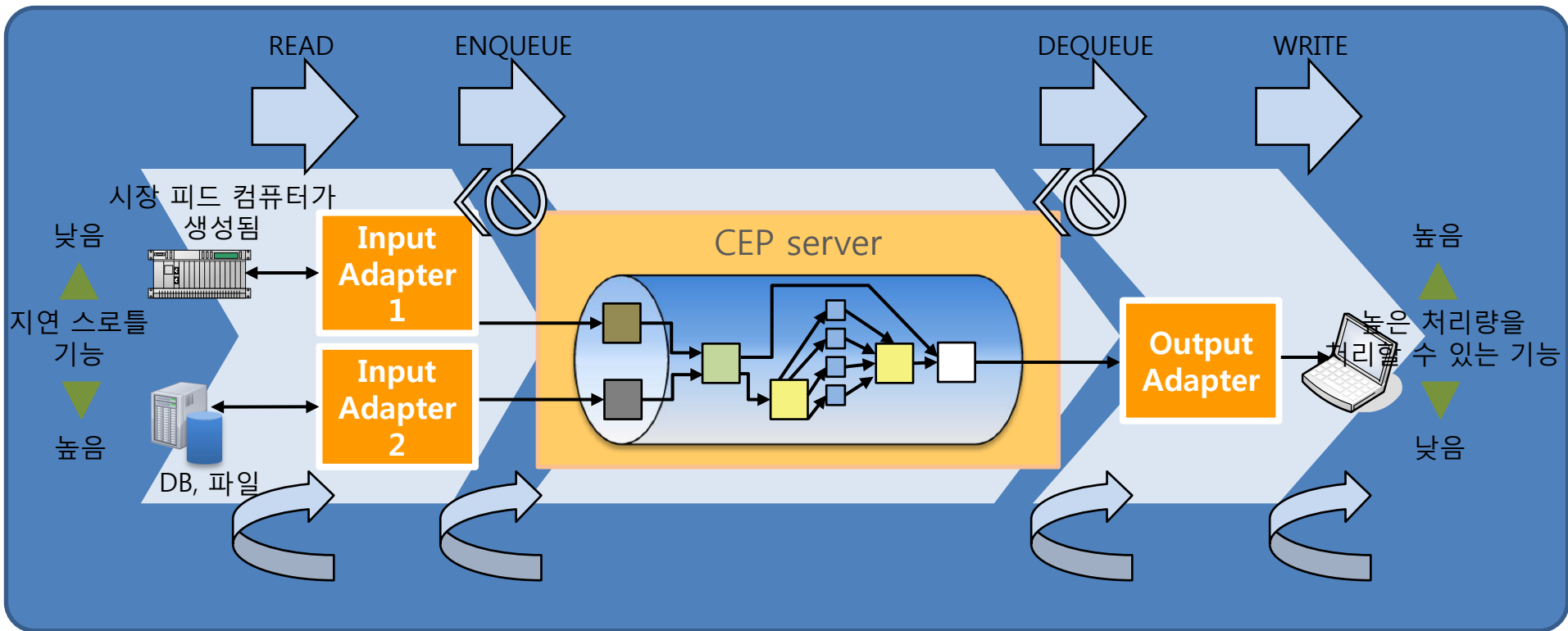
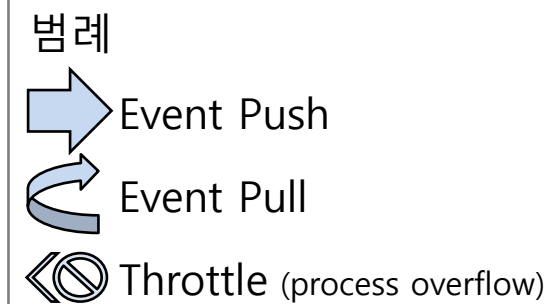


# Application Component

## Adapters

### Event Stream Adapters : Push vs. Pull Model

- Interaction points support Push and Pull
- Consumer can throttle input



## Adapter Development Tasks

Determine Event Type, Event Model, and Event Kind



Choose Appropriate Adapter Base Class



Design Flow Control Logic for READ and WRITE



Design ENQUEUE or DEQUEUE Flow Control Interaction



Design AdapterFactory Object

# Application Component

## Development Model



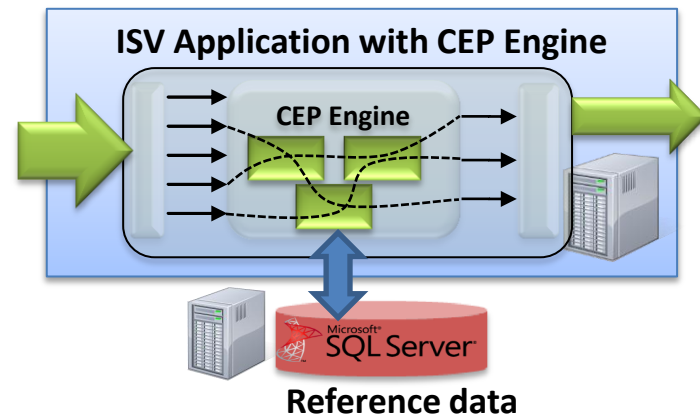
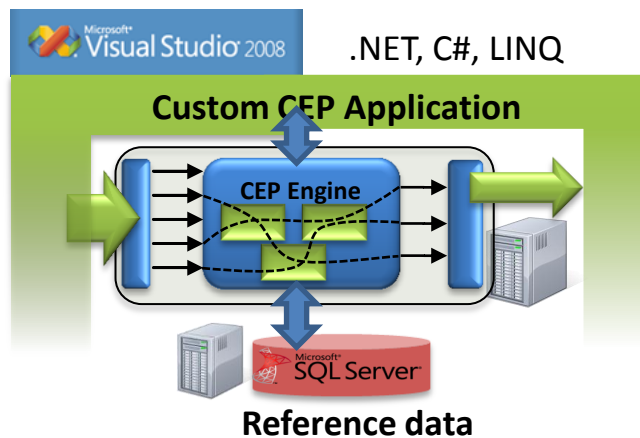
## StreamInsight Development Two Models

Implicit Server	Explicit Server
Easiest	Most Flexible
Hides most complexity	Provide complete control of StreamInsight application, Development Environment
Allows developers to focus on query logic	Allows for reuse Queries, Adapters, Event Types, Third-party Query Templates
Query is automatically hosted	Code creates all objects and registers them into server
Stored in memory, not on stable storage (disk)	Server can store metadata in memory or persist to disk
	Server can run locally or remotely via Web service

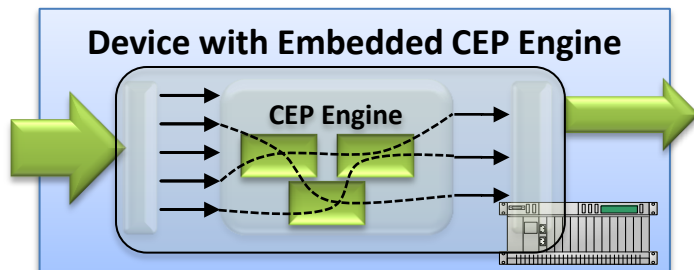
# Application Component Deployment Model

## CEP Deployment Scenarios

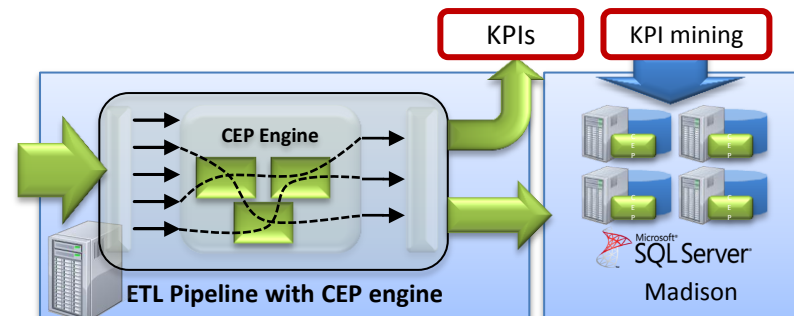
**Scenario 1: Custom CEP Application Dev**    **Scenario 2: Embed CEP in Application**



**Scenario 3: CEP Enabled Device**



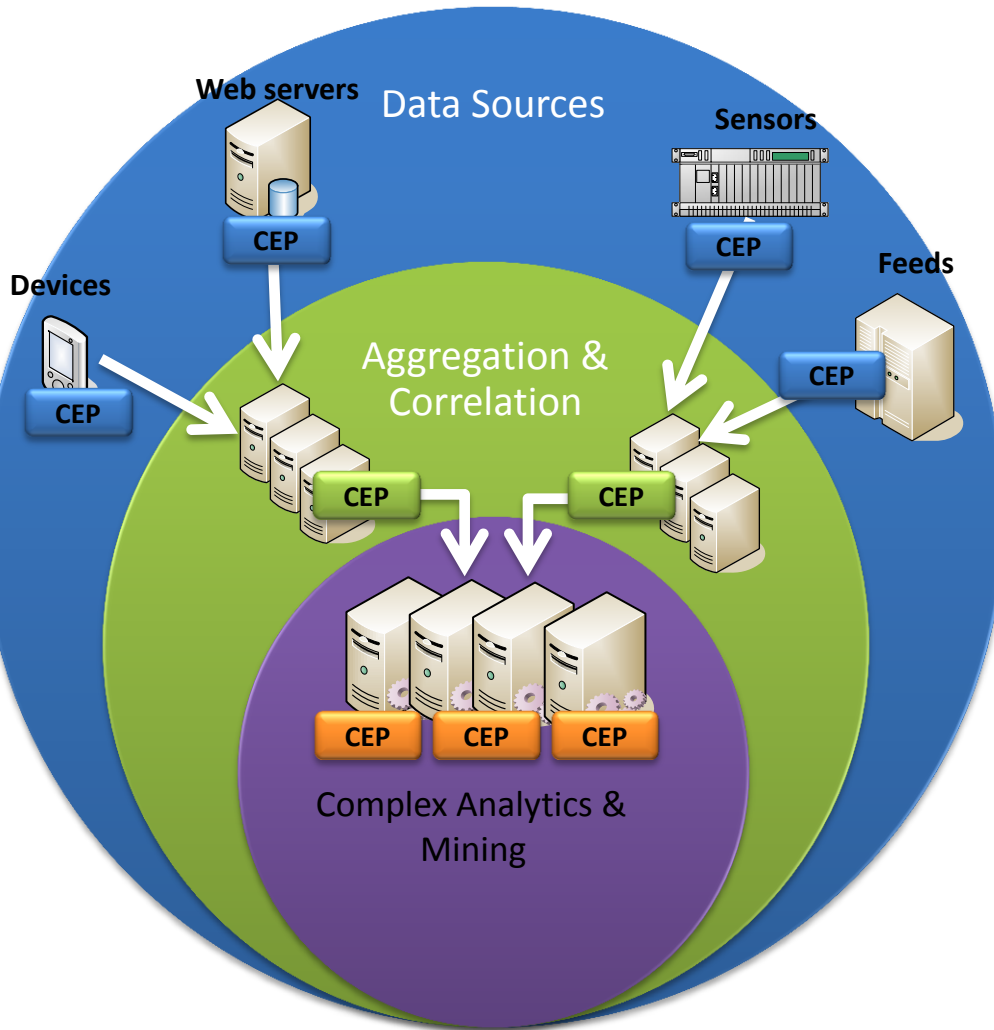
**Scenario 4: Operational Intelligence w/ CEP**





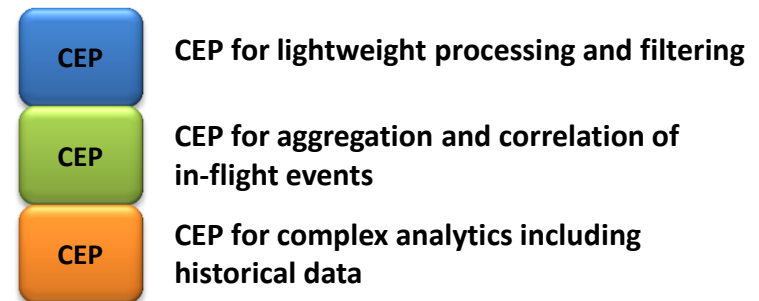
# Application Component Deployment Model

## CEP Deployment alternatives



Event processing engines are deployed at multiple places on different scales

- At the edge – close to the data source
- In the mid-tier – consolidate related data sources
- In the data center – historical archive, mining, large scale correlation.



# Application Component

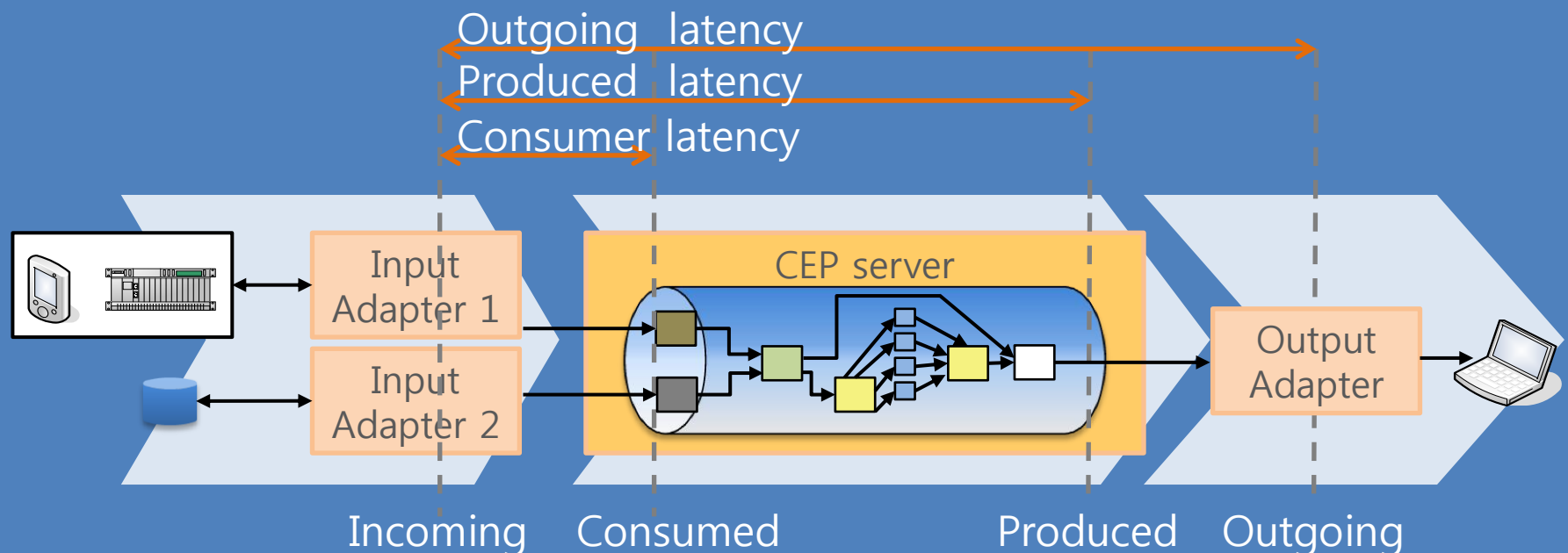
## Monitoring & Event Flow Debugger



### Monitoring StreamInsight

- Need to track : Overall health of system, Query performance
- Diagnostic views using ManagementService API
  - Use URIs for resource naming
  - GetDiagnosticView(), SetDiagnosticSettings(), ClearDiagnosticSettings()
  - Can be accessed via PowerShell

### Query Monitoring Points



# Application Component Monitoring & Event Flow Debugger

## Event Flow Debugger Demo (TechEdUS2010)

The screenshot displays the Microsoft StreamInsight Event Flow Debugger interface. The main window shows a flow diagram for a query named "MarketMonitorQuery". The flow consists of the following components:

- QuoteInput Input Adapter
- QuoteInput\_CleanseInput Cleanse Input
- Grouping.1.1 Group-0개의 그룹 (highlighted with a dashed blue box)
- 그룹 입력 Input to Group(접힘) (highlighted with a dashed blue box)
- Grouping.1.1.Aggregate.1.1 Aggregate(접힘) (highlighted with a dashed blue box)
- Grouping.1.1\_GroupUnion Group Union (highlighted with a dashed blue box)
- OutputAdapter\_Cleanse Cleanse

A dialog box titled "기록..." (Log...) is open over the flow diagram, containing a text input field and a "중지" (Stop) button. The left sidebar shows the project structure, including "MarketMonitorQuery (실행 중)" (MarketMonitorQuery (Running)). The bottom status bar indicates a zoom level of 100%.

# Application Component Monitoring & Event Flow Debugger

## Event Flow Debugger Demo (TechEdUS2010)

The screenshot displays the Microsoft StreamInsight Event Flow Debugger interface. The left pane shows a tree view of the application components, including '이벤트' (Events), '쿼리 관리자' (Query Manager), '복구 관리자' (Recovery Manager), and '쿼리' (Query). The main pane shows the 'MarketMonitorQuery' component, which is currently selected. Below the component name, there is a table of events. The table has columns: EventKind, StartTime, EndTime, Ask, AskSize, AssetClass, Bid, BidSize, Country, Last, Market, Region, Ticker, and Volume. The first row is highlighted in orange and represents an 'Insert' event for a stock.

EventKind	StartTime	EndTime	Ask	AskSize	AssetClass	Bid	BidSize	Country	Last	Market	Region	Ticker	Volume
Insert	2012-04-02 21:07:53.0040435	2012-04-02 21:07:53.0040436	4	984	Stock	6	860	US	12	NASC	NA	MSF	880

Below the table, it indicates '1개의 이벤트' (1 event). Below that, the 'QuoteInput\_CleanseInput' component is shown, which is a 'Cleanse Input' component. It also displays a table of events, with the last row highlighted in green, representing a 'Cti' event.

EventKind	StartTime	EndTime	Ask	AskSize	AssetClass	Bid	BidSize	Country	Last	Market	Region	Ticker	Volume
Insert	2012-04-02 21:07:52.9980432	2012-04-02 21:07:52.9980433	92	213	Derivative	46	176	US	44	NYSE	NA	IBM	744
Cti	2012-04-02 21:07:52.9980433												
Insert	2012-04-02 21:07:52.9990432	2012-04-02 21:07:52.9990433	10	187	Derivative	16	669	US	44	NYSE	NA	IBM	665
Cti	2012-04-02 21:07:52.9990433												
Insert	2012-04-02 21:07:53.0040435	2012-04-02 21:07:53.0040436	4	984	Stock	6	860	US	12	NASC	NA	MSF	880
Cti	2012-04-02 21:07:53.0040436												

Below the table, it indicates '6개의 이벤트' (6 events). At the bottom, a 'Grouping.1.1' component is shown, which is a 'Grouping' component. It indicates 'Group-8개의 그룹' (Group-8 groups).

# Usage Example

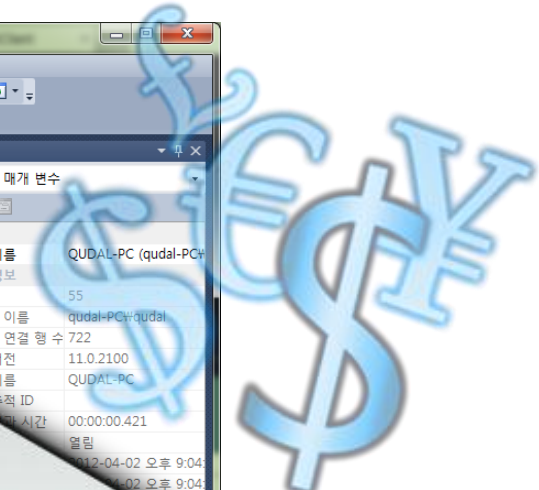
## Capital Markets

### Demo (TechEdUS2010)

	timestamp	ticker	assetclass	market	country	region	avglast
715	2012-04-02 12:02:59.000	IBM	Derivative	NASDAQ	US	NA	47,1170212765957
716	2012-04-02 12:02:59.000	IBM	Derivative	NYSE	US	NA	50,2685185185185
717	2012-04-02 12:02:59.000	IBM	Stock	NASDAQ	US	NA	48,1666666666667
718	2012-04-02 12:02:59.000	IBM	Stock	NYSE	US	NA	47,6923076923077
719	2012-04-02 12:02:59.000	MSFT	Derivative	NASDAQ	US	NA	49,1386138613861
720	2012-04-02 12:02:59.000	MSFT	Derivative	NYSE	US	NA	50,6696428571429
721	2012-04-02 12:02:59.000	MSFT	Stock	NASDAQ	US	NA	49,3211009174312
722	2012-04-02 12:02:59.000	MSFT	Stock	NYSE	US	NA	45,7027027027027

준비

이름  
연결 이름입니다.



# Usage Example Capital Markets

## Demo (TechEdUS2010)

Microsoft SQL Server Enterprise Manager showing a query execution in the Query Analyzer window. The query is:

```
select * from MarketSnapshots
```

The results table shows the following data:

timestamp	ticker	assetclass	market	country	region	avglast
2012-04-02 12:02:59.000	IBM	Derivative	NASDAQ	US	NA	47.1170212765957
2012-04-02 12:02:59.000	IBM	Derivative	NYSE	US	NA	50.2685185185185
2012-04-02 12:02:59.000	IBM	Stock	NASDAQ	US	NA	48.1666666666667
2012-04-02 12:02:59.000	IBM	Stock	NYSE	US	NA	47.6923076923077
2012-04-02 12:02:59.000	MSFT	Derivative	NASDAQ	US	NA	49.1386138613861
2012-04-02 12:02:59.000	MSFT	Derivative	NYSE	US	NA	50.6696428571429
2012-04-02 12:02:59.000	MSFT	Stock	NASDAQ	US	NA	49.3211009174312
2012-04-02 12:02:59.000	MSFT	Stock	NYSE	US	NA	45.7027027027027

WCF Duplex CEP Server console output showing log messages and data rows:

```

Polling WCF Duplex Server started...
Press <ENTER> to stop.
Creating CEP Server
Creating CEP Application
Registering LINQ query template
Registering Adapter Factories
Registering bound query
Start query
CTI 2012-04-02 11:53:48.0000000
INSERT 2012-04-02 11:53:48.0000000 Stock 35.25 354.5 43.5 633.25
INSERT 2012-04-02 11:53:48.0000000 Derivative 42.1428571428571
643.428571428571 US NYSE NA IBM
INSERT 2012-04-02 11:53:48.0000000 Derivative 45.8571428571429
7142857143 400.428571428571 US NASDAQ NA IBM
INSERT 2012-04-02 11:53:48.0000000 Stock 66.25 568.75 48.75 642.25
INSERT 2012-04-02 11:53:48.0000000 Derivative 40.5 460.75 53
INSERT 2012-04-02 11:53:48.0000000 Derivative 39 516 63.5
INSERT 2012-04-02 11:53:48.0000000 Stock 31.5 470.5 72 539.25
INSERT 2012-04-02 11:53:48.0000000 Stock 57 493.666666666667
33 US NYSE NA MSFT
CTI 2012-04-02 11:53:49.0000000
INSERT 2012-04-02 11:53:49.0000000 Derivative 44.2142857142857
8571428571 594.285714285714 US NYSE NA IBM
INSERT 2012-04-02 11:53:49.0000000 Stock 66.4444444444444 583.5555
4444444444 US NYSE NA IBM
INSERT 2012-04-02 11:53:49.0000000 Derivative 45.1875 405.9375
IBM
INSERT 2012-04-02 11:53:49.0000000 Derivative 42.2857142857143
2857142857 550.142857142857 US NASDAQ NA MSFT
INSERT 2012-04-02 11:53:49.0000000 Stock 44.4615384615385 469.9230
8461538462 485.384615384615 US NASDAQ NA IBM
INSERT 2012-04-02 11:53:49.0000000 Stock 42.7272727272727 424
7272727273 US NASDAQ NA MSFT
INSERT 2012-04-02 11:53:49.0000000 Derivative 38.5833333333333
US NYSE NA MSFT
INSERT 2012-04-02 11:53:49.0000000 Stock 53.8333333333333 518.3333
NYSE NA MSFT
CTI 2012-04-02 11:53:50.0000000
INSERT 2012-04-02 11:53:50.0000000 Stock 55.7142857142857 658.5
4285714286 US NYSE NA IBM
INSERT 2012-04-02 11:53:50.0000000 Stock 44.2941176470588 473.8823
2941176471 544 US NASDAQ NA IBM
INSERT 2012-04-02 11:53:50.0000000 Derivative 41.3333333333333
5714285714 608.190476190476 US NYSE NA IBM
INSERT 2012-04-02 11:53:50.0000000 Derivative 40.8 450.04 41.6
INSERT 2012-04-02 11:53:50.0000000 Stock 51.6 454.7 50 447.25
INSERT 2012-04-02 11:53:50.0000000 Derivative 37.1739130434783
0869565217 523.695652173913 US NASDAQ NA MSFT
INSERT 2012-04-02 11:53:50.0000000 Stock 56.4090909090909 488.5909
45454545 599.227272727273 US NYSE NA MSFT
INSERT 2012-04-02 11:53:50.0000000 Derivative 45.0869565217391
457.391304347826 US NYSE NA MSFT
CTI 2012-04-02 11:53:51.0000000

```

# Usage Example

## Capital Markets



### Demo (TechEdUS2010)

```
WCF Duplex CEP Server
Polling WCF Duplex Server started...
Press <ENTER> to stop.
Creating CEP Server
```

#### StreamInsight Demo StreamInsight Demo

#### TechEd 2010

Start StreamInsight Engine Listen to Output Adapter Start Output to SQL Table Stop StreamInsight Engine

```
New event: CTTI2012-04-02 11:54:26.0000000.
New event: INSERT2012-04-02 11:54:25.0000000Stock 49.4870466321244509.70120898100249.713298791019 499.24179620034550.740932642487 486.474956822107US
NYSE NA MSFT .
New event: INSERT2012-04-02 11:54:25.0000000Derivative 50.9238249594814490.38411669367949.515397082658 498.06807131280448.7698541329011
503.293354943274US NASDAQ NA IBM.
New event: INSERT2012-04-02 11:54:25.0000000Derivative 48.9523809523809503.99294532627948.6860670194004507.97178130511550.4585537918871477.57671957672
US NASDAQ NA MSFT .
New event: INSEPT2012-04-02 11:54:25.0000000Stock 49.0945512820513502.25641025641 50.3076023076023513 500.35807435048 1858074358074406 41666666666671US
```

#### Stock Prices

Time	MSFT	IBM
11:53:58	49.0	47.5
11:54:00	51.5	44.5
11:54:02	51.8	46.5
11:54:04	51.5	48.0
11:54:06	50.0	51.5
11:54:08	49.0	49.5
11:54:10	50.0	50.0
11:54:12	50.5	50.0
11:54:14	51.0	50.5
11:54:16	51.5	49.0
11:54:18	51.8	49.5
11:54:20	51.5	48.0
11:54:22	50.0	49.0
11:54:24	48.5	48.5

```
tock 35.25 354.5 43.5 633.25
rivative 42.1428571428571
a IBM
rivative 45.8571428571429
$ NASDAQ NA IBM
tock 66.25 568.75 48.75 642.25
rivative 40.5 460.75 53
rivative 39 516 63.5
tock 31.5 470.5 72 539.25
tock 57 493.666666666667

rivative 44.2142857142857
$ NYSE NA IBM
tock 66.4444444444444 583.5555
BM
rivative 45.1875 405.9375

rivative 42.2857142857143
$ NASDAQ NA MSFT
tock 44.4615384615385 469.9230
$ NASDAQ NA IBM
tock 42.7272727272727 424
SFT
rivative 38.5833333333333

tock 53.8333333333333 518.3333

tock 55.7142857142857 658.5
BM
tock 44.2941176470588 473.8823
a IBM
rivative 41.3333333333333
$ NYSE NA IBM
rivative 40.8 450.04 41.6
tock 51.6 454.7 50 447.25
rivative 37.1739130434783
$ NASDAQ NA MSFT
tock 56.4090909090909 488.5909
$ NYSE NA MSFT
rivative 45.0869565217391
a MSFT
```

# Usage Example



## Manufacturing:

- Sensor on plant floor
- React through device controllers
- Aggregated data
- 10,000 events/sec

## Web Analytics:

- Click-stream data
- Online customer behavior
- Page layout
- 100,000 events /sec

## Financial Services:

- Stock & news feeds
- Algorithmic trading
- Patterns over time
- Super-low latency
- 100,000 events /sec

## Power, Utilities:

- Energy consumption
- Outages
- Smart grids
- 100,000 events/sec



## Asset Instrumentation for Data Acquisition, Subscriptions to Data Feeds

Data Stream



Visual trend-line and KPI monitoring  
Batch & product management  
Automated anomaly detection  
Real-time customer segmentation  
Algorithmic trading  
Proactive condition-based maintenance

Data Stream

Stream Data Store & Archive

Asset Specs & Parameters

Lookup

Event Processing Engine

- Threshold queries
- Event correlation from multiple sources
- Pattern queries



# Competitive Landscape

Progress – Apama	IBM – Open ESB
West Global – Total Insight	Coral8
Bristol Technology – SenActive	GmbH – Real Time Monitoring
Streambase	Agent Logic
Inetco	Tibco - BusinessEvents
Oracle – Fusion Middleware	Event Zero
Syndera	Rulecare
LG CNS Event Pro	Sybase Aleri Event Stream Processor

Industry Forum: <http://complexevents.com>

# Resources

- StreamInsight Website
  - <http://www.microsoft.com/sqlserver/en/us/solutions-technologies/business-intelligence/complex-event-processing.aspx>
- StreamInsight Books Online
  - [http://msdn.microsoft.com/ko-kr/library/hh750619\(v=sql.10\).aspx](http://msdn.microsoft.com/ko-kr/library/hh750619(v=sql.10).aspx)
- StreamInsight Forums
  - <http://social.msdn.microsoft.com/Forums/en-US/streaminsight/threads>
- StreamInsight Official blog
  - <http://blogs.msdn.com/b/streaminsight/>
- SQL Server 2008 R2 Update for Developers Training Kit (May 2011 Update)
  - <http://www.microsoft.com/download/en/details.aspx?displaylang=en&id=16281>



# SQL Unplugged 2012 : 쇼케이스