The Future of the Raima Embedded DBMS

Wayne Warren & Randy Merilatt
Raima Inc.

Quick Tour of Raima Product History

- 1984: db_VISTA: Single-user library, DDLP, Initdb.
- 1985: Portable multi-user library using lock file and guard file.
- 1987: Multi-user library with lock manager.
  - SQL-like query/report writing tool.
- 1989: Multi-tasking library for Windows & segmented memory management.
- 1993: RDS (aka Velocis, RDM Server) - Client/server DBMS with record-level locking.
  - “Full" SQL.

Quick Tour of Raima Product History

- 1994-1999: Various features added to RDMs.
  - True OS threading.
  - Hot database backup.
- 1999-2009: Some significant new features.
  - RDMs: dynamic DDL, replication to 3rd party DBMSs, SQL extended join syntax + triggers.
  - RDMe: inmemory, more RTOS platform support, replication to RDMs & ODBC 3rd party DBMSs.

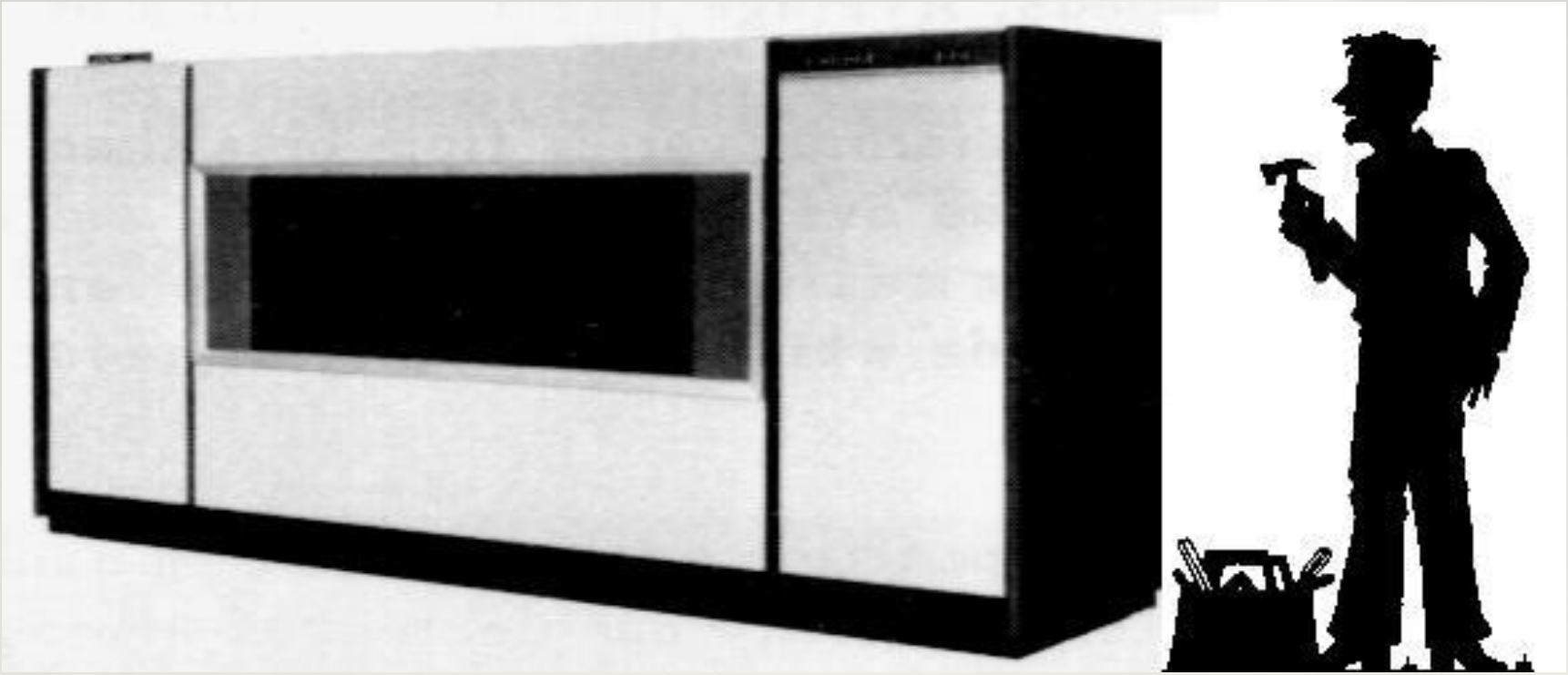
Quick Tour of Raima Product History

- 2010: RDMe 10 with …
  - Transactional File Server.
  - New replication.
    - to RDM Embedded.
    - to 3rd party with aggregation.
  - Improved in-memory storage.
  - Mirroring.
  - Read-only transactions.
  - Distributed queries.
  - Two-phase commit support.

Quick Tour of Hardware Evolution

- ~1984: 16-bit, 64k memory maximum. 10mb hard drives.
- ~1985: 640k memory available through segmented programming.
- Weak LANs. Multi-User through Unix.
- 1990s: 32-bit processors (no more segmented memory problems).
- Rise of Internet & LAN.
  - Strong server computers, weak clients => client/server.
- Memory & disk capacities increasing exponentially (still are).
- CPU speeds increase to 3GHz+.
- ~2000+: Multi-core processors.

The State-Of-The-Art Has Changed



Univac Fastrand III Drum Storage (circa 1973)
~500 mb

1984 DB_VISTA Design Goals

- Control use of memory.
- Optimize for performance over disk space.
- C programmer-friendly.
- Portability (via #ifdef's).

1991 RDS Design Goals

- Multi-threading: make sure CPU is busy.
- Client/Server architecture.
- Improved portability (via resource mgr).

RDMe 10 Design Goals in 2010

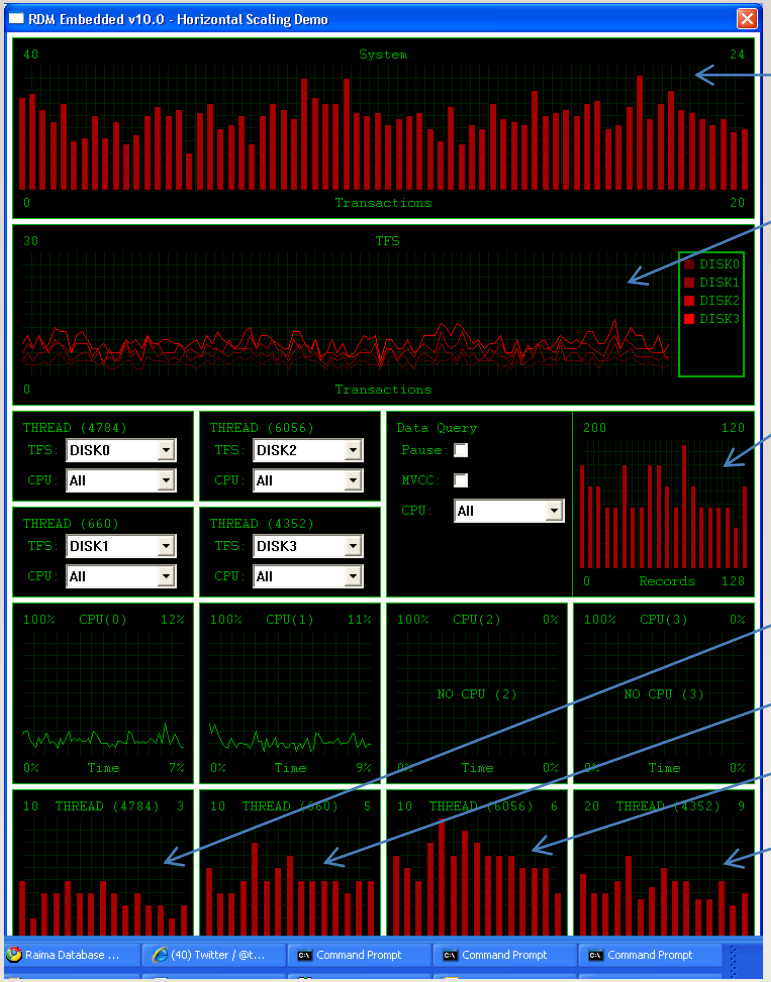
- Utilize strong client computers.
- Utilize operating systems.
  - Good file I/O and caching.
  - Good virtual memory.
  - Reliable file system.
- Utilize network.
- Utilize multi-core CPU.
  - Concurrency is not the same as parallelism.
  - CPU speed has peaked.
  - Multi-CPU computers is becoming the norm.
- Make easier to use.

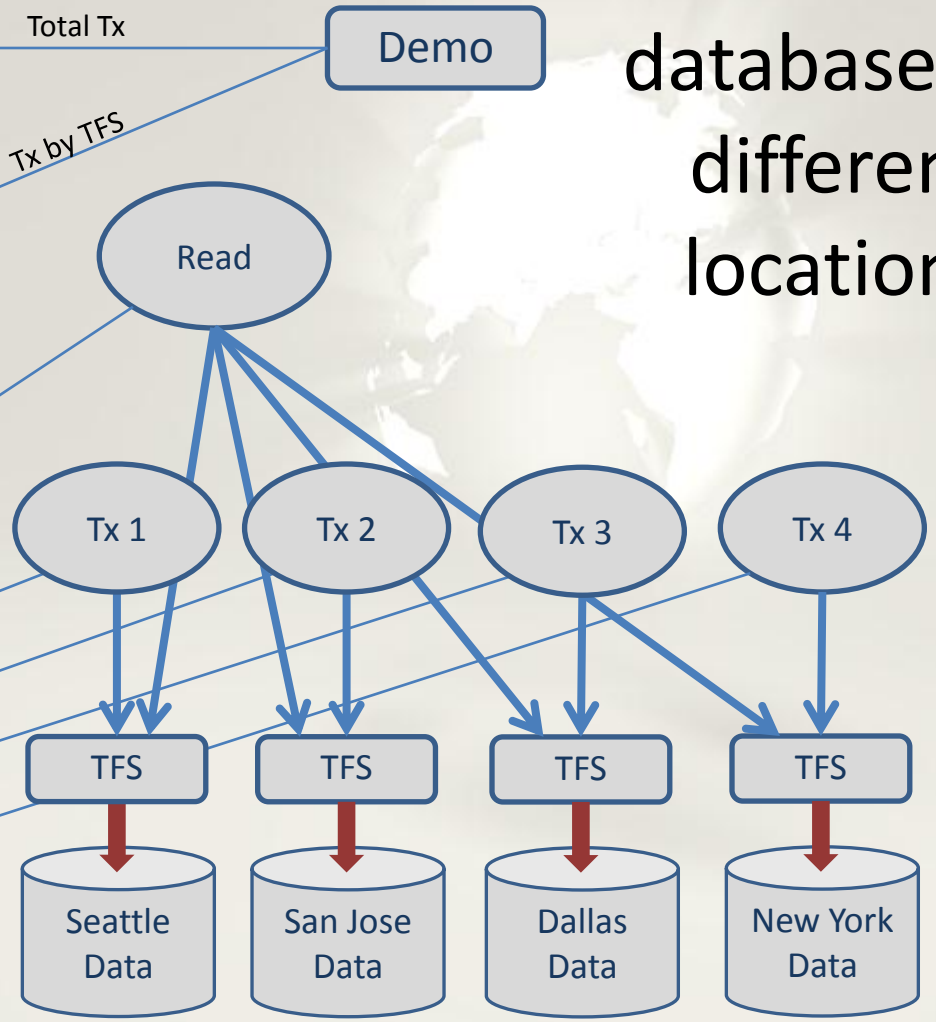
Distributed Database Demo

Demonstration of the following features:

- Access to databases in different locations.
- Distributed database query.
- Read-Only-Transactions.
  - AKA Multi-Version-Concurrency-Control (MVCC)

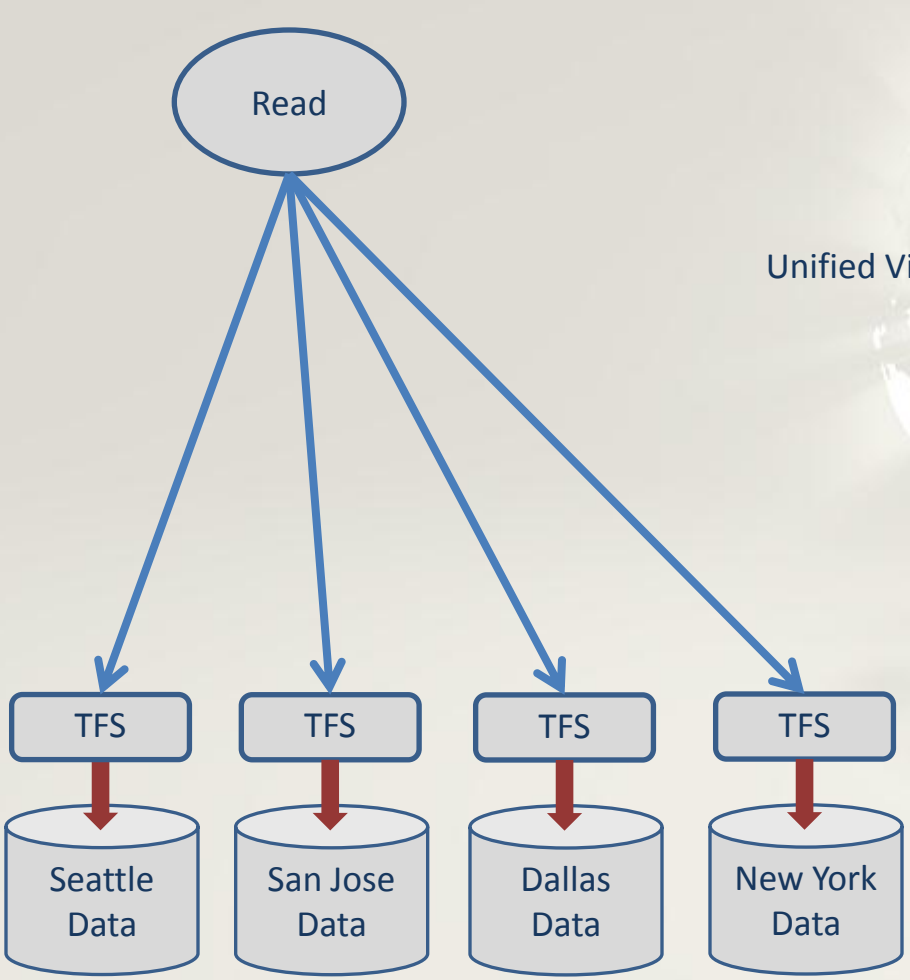
DISTRIBUTED DATABASE DEMO

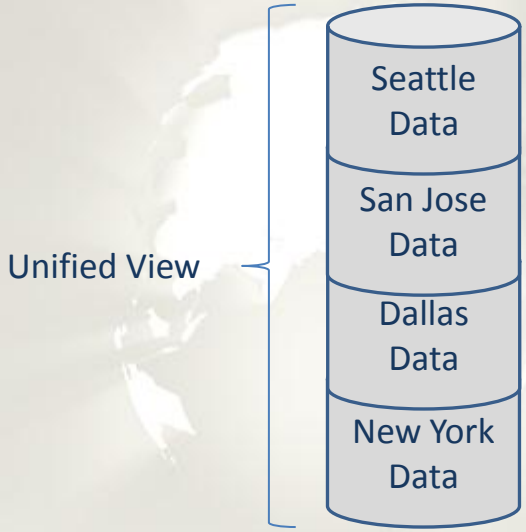




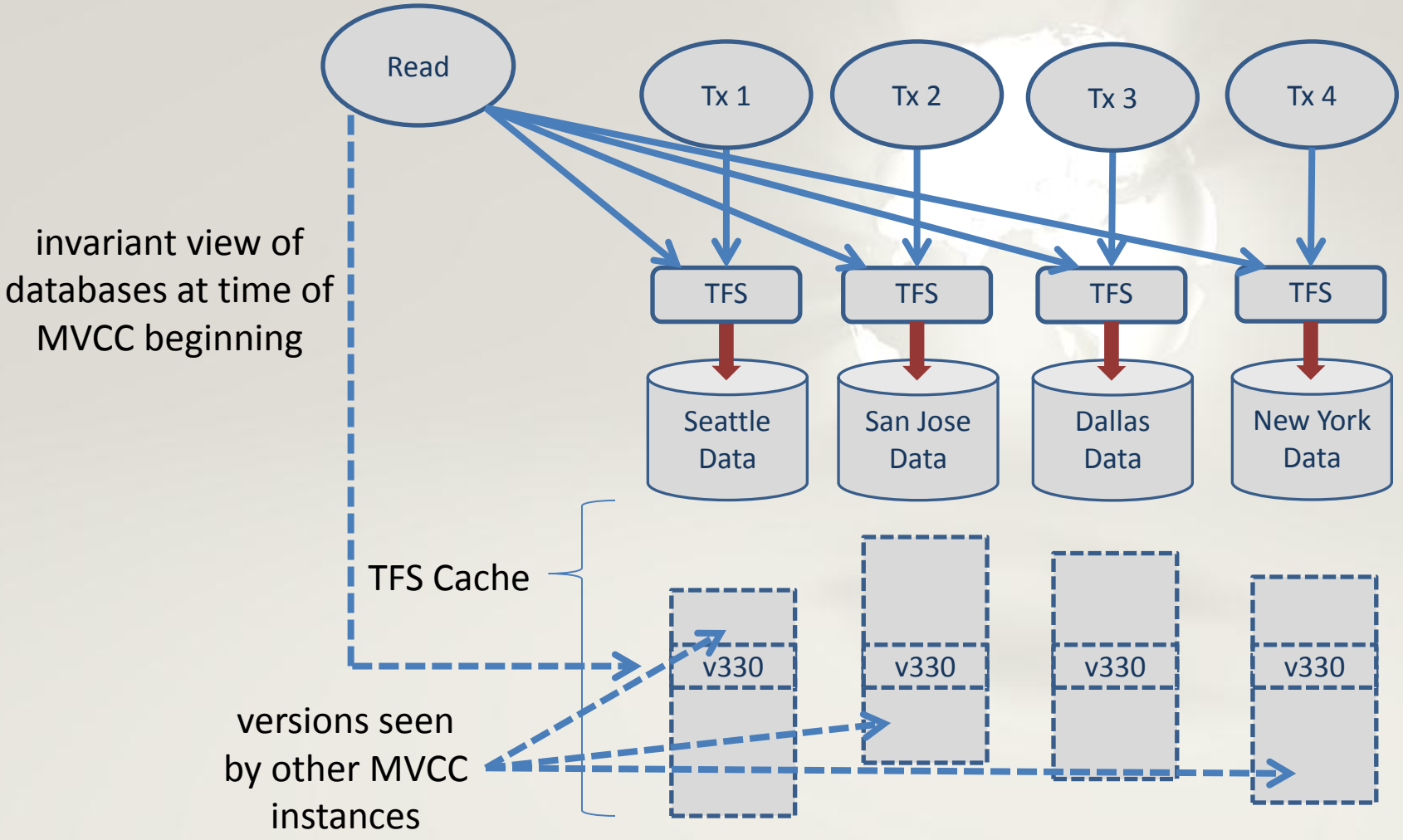
Access to databases in different locations

Distributed database query



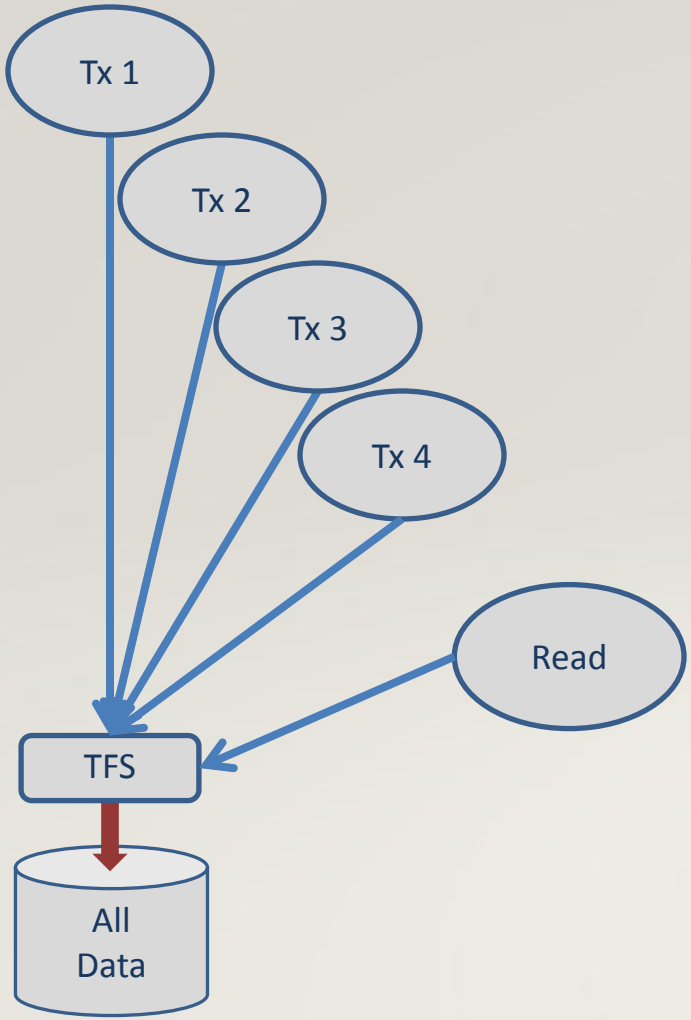


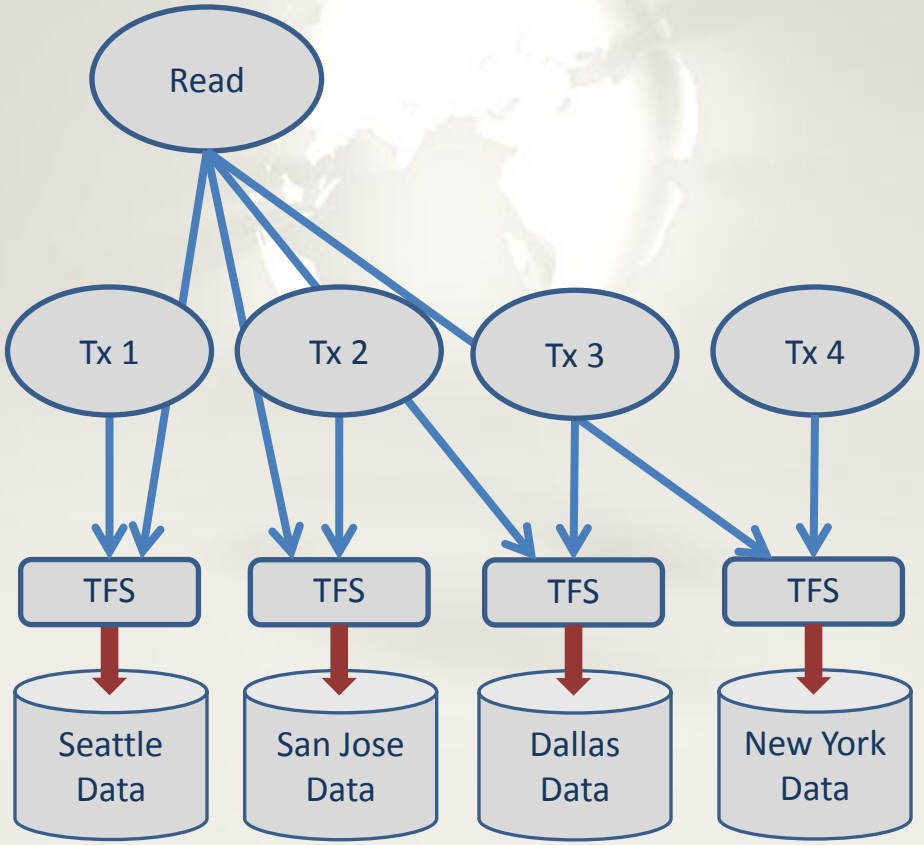
Read-Only-Transactions



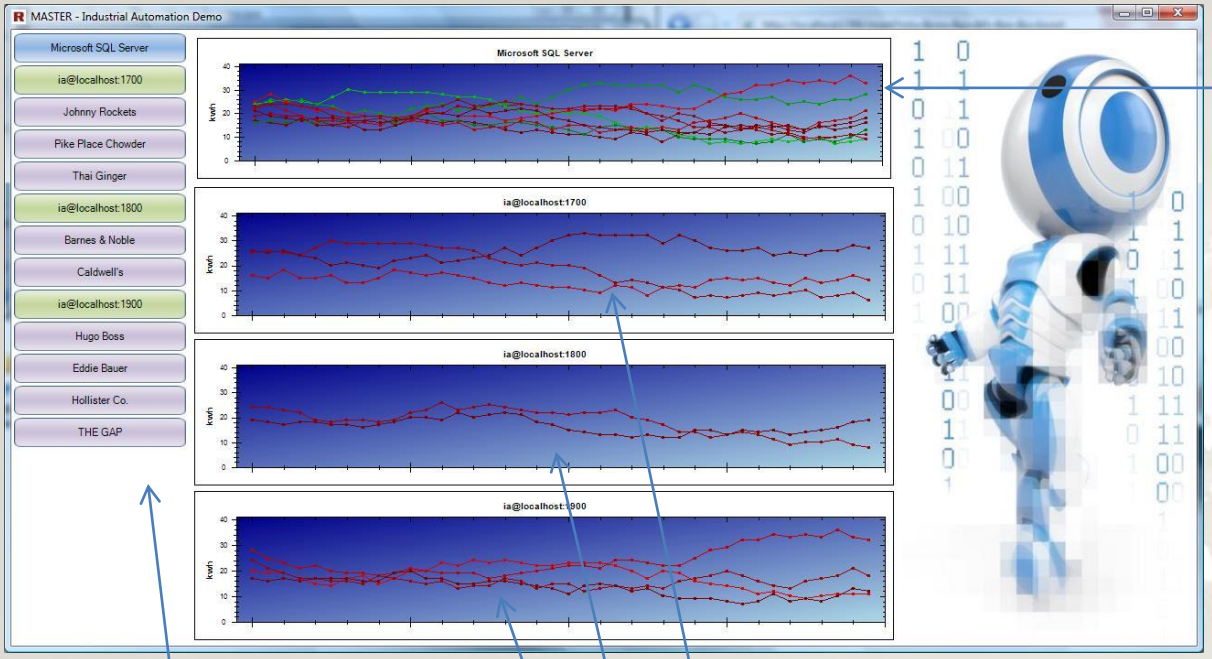
Distributed Database Demo

Scaling up performance through distribution





DATAFLOW DEMO



Graph of Unit readings in SQL Server database

Summary of all Units

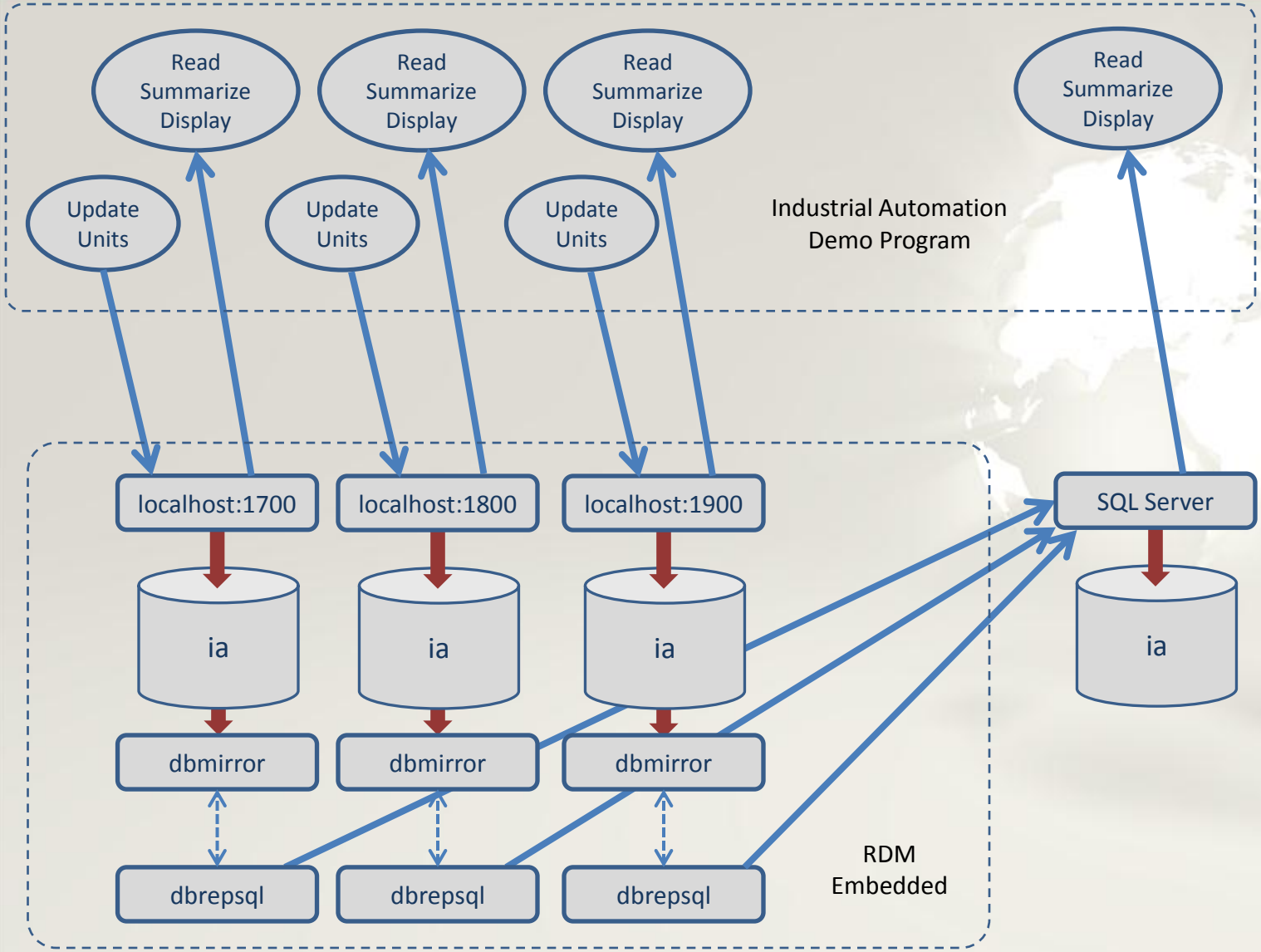
Graph of Unit readings in first slave database

Second slave database

Third slave database

Three master databases are replicated into the MS SQL Server database

DATAFLOW DEMO



Next Version of RDM Embedded?

- Embedded SQL.
- Enhanced Interoperability.
  - JDBC
  - ADO.NET
  - ODBC
- Enhanced HA Support.
- Hash Indexing.
- Browser-based Monitor/Administrator.

An Embedded SQL DBMS?

Not very likely…

- The SQL standard is huge and complex.
- SQL is designed for enterprise not embedded environments.
- SQL typically requires a lot of computing horsepower.
- SQL is not particularly programmer-friendly.

An Embedded SQL DBMS?

On the other hand…

- An abridged SQL with only those features useful in embedded applications can be produced.
- Embedded computing environments are less resource constrained today.
- SQL is well-understood by many developers.
  - The ability to use SQL on the smallest devices up to the enterprise is very appealing.

Unnecessary SQL Features

- Security.
- Views.
- Table/column constraint checking.
- Triggers.
- Dynamic DDL.

RDMe SQL for Embedded Applications

- Ability to run disk-less.
  - Use of RDMe in-memory feature.
  - Catalog info in generated C module.
  - SQL statements in pre-compiled stored procedures in generated C modules.

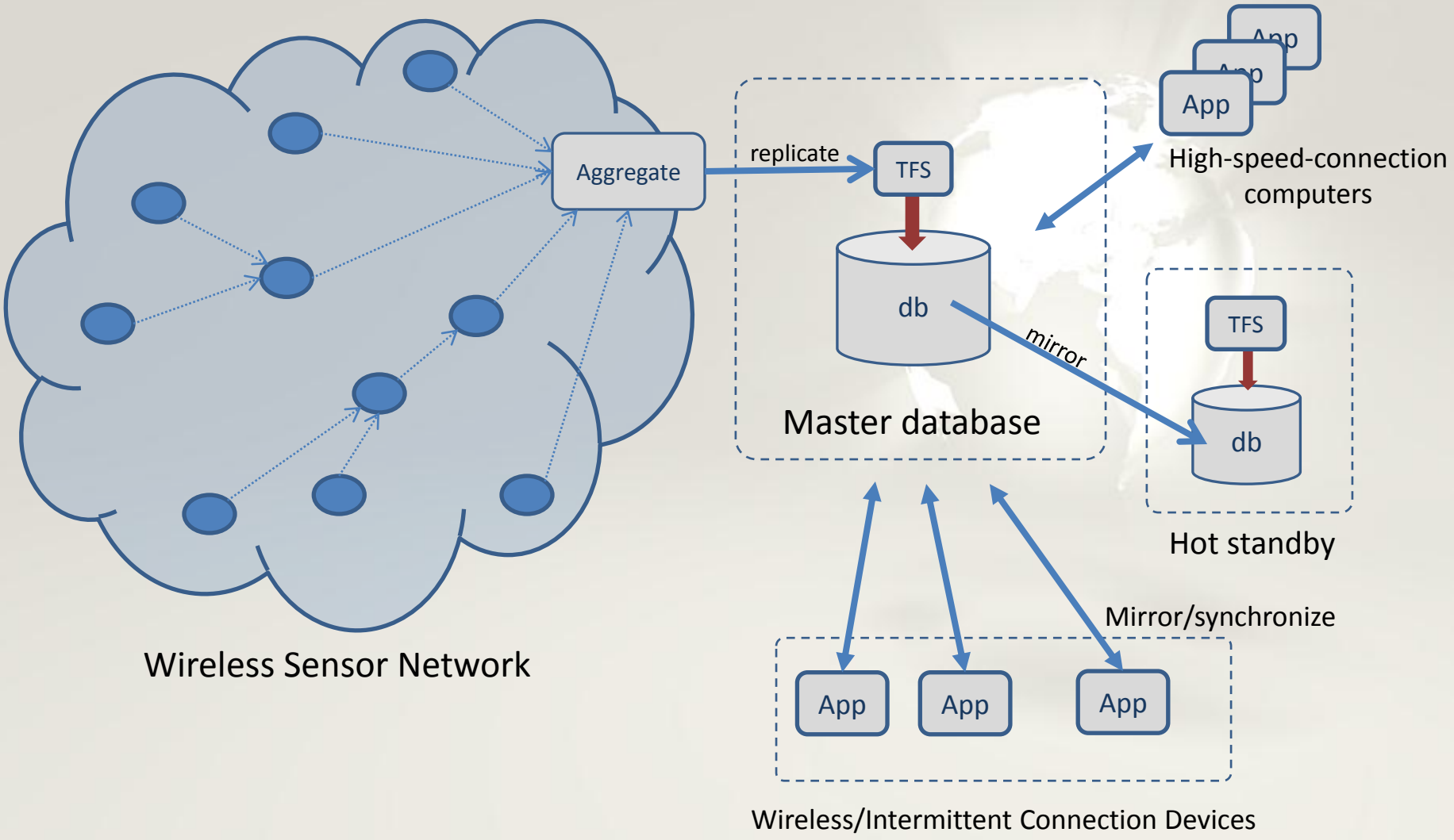
RDMe SQL for Embedded Applications

- C-based user-defined functions.
- C-based virtual tables.
  - Use of **limit** clause on **select** statement for unending real-time virtual tables.
- More details: click on 1st link (…"Elephant"…) under "Just Published" on Raima home page.
- Currently under development.

Future Embedded DBMS Needs

- Interoperability
  - Small computer to large
  - Network connectivity
  - VxWorks, Unix, Windows, Linux, Android
- High Availability
  - From anywhere
  - With any device
  - Data redundancy/distribution
  - Failure recovery

Future Embedded DBMS Needs



Future Embedded DBMS Needs (cont)

- Partitioned/Distributed Databases.
- Granular Locking.
- Performance.
- Predictability.
  - Decreased variance in response times.
- Security.
- Certifications.

Questions?

For More Information

- Download the RDMe Core free SDK at: http://www.raima.com/downloads/
- Contact sales@raima.com to setup your evaluation of RDMe HA, dataFlow, Distributed or all.
- Start solving real world problems with RDM Embedded.