



IBM Almaden Research Center

Information Integration Isn't Simple

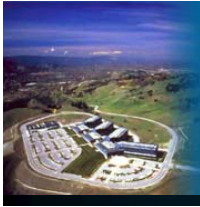
Laura Haas
IBM Distinguished Engineer
Director, Computer Science
Almaden Research Center



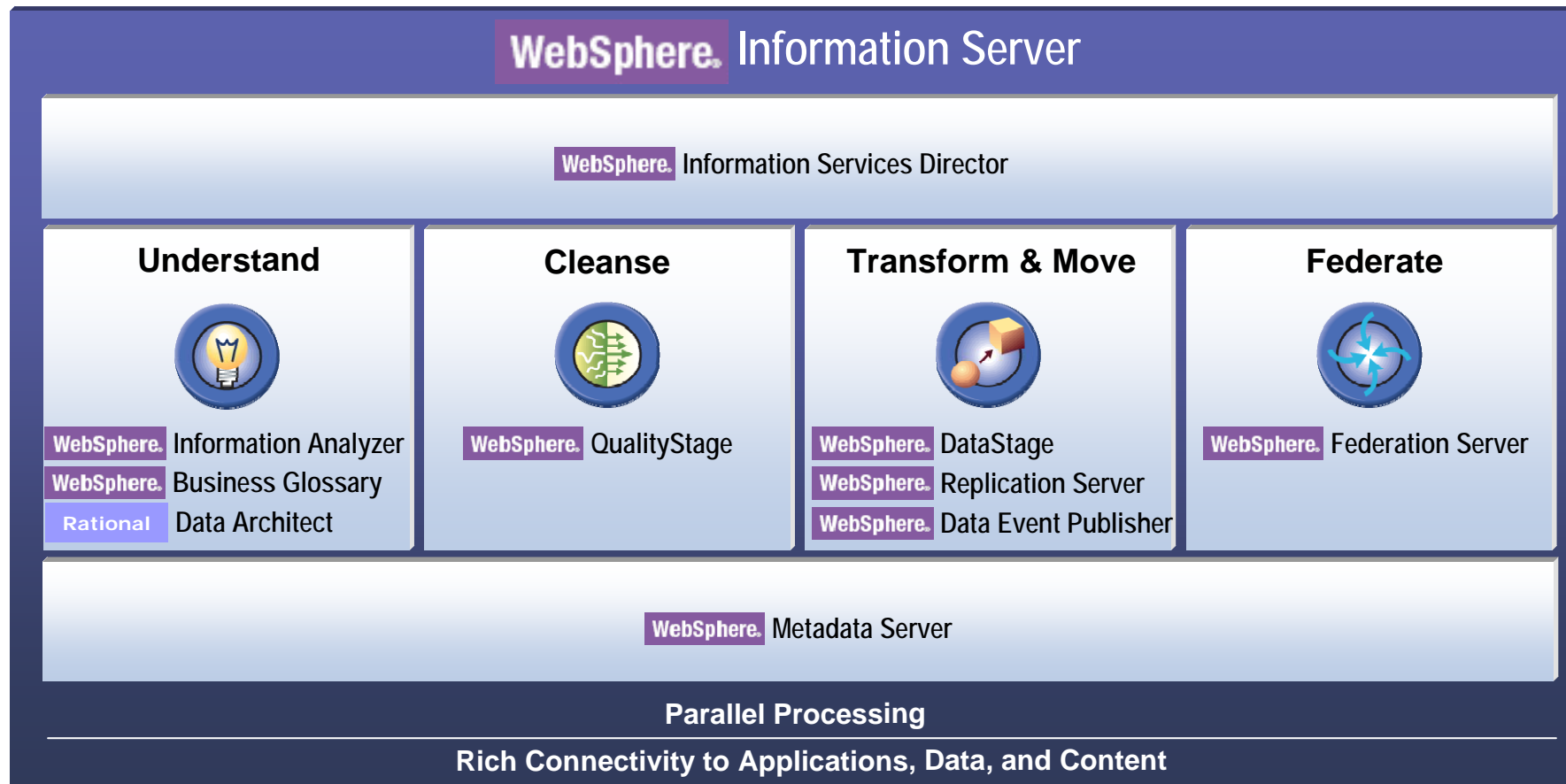


Integration is a Process

- Understand
 - Find appropriate data
 - Analyze data
 - Discover relationships
- Standardize
 - Design target schema
 - Define representations and terminology
 - Identify inconsistencies and incompleteness
 - Resolve entities (deduplicate)
 - Define how to correct or tolerate
- Specify
 - Determine and configure execution engine
 - Define mappings (declaratively or procedurally)
 - Create code/script/other to direct execution
- Execute
 - Materialize (ETL, Replication, Caching, ...)
 - Federate (EII)
 - Search
 - Process integration (EAI, BPI, EJBs, queues, ...)
- Overlapping, not completely ordered tasks

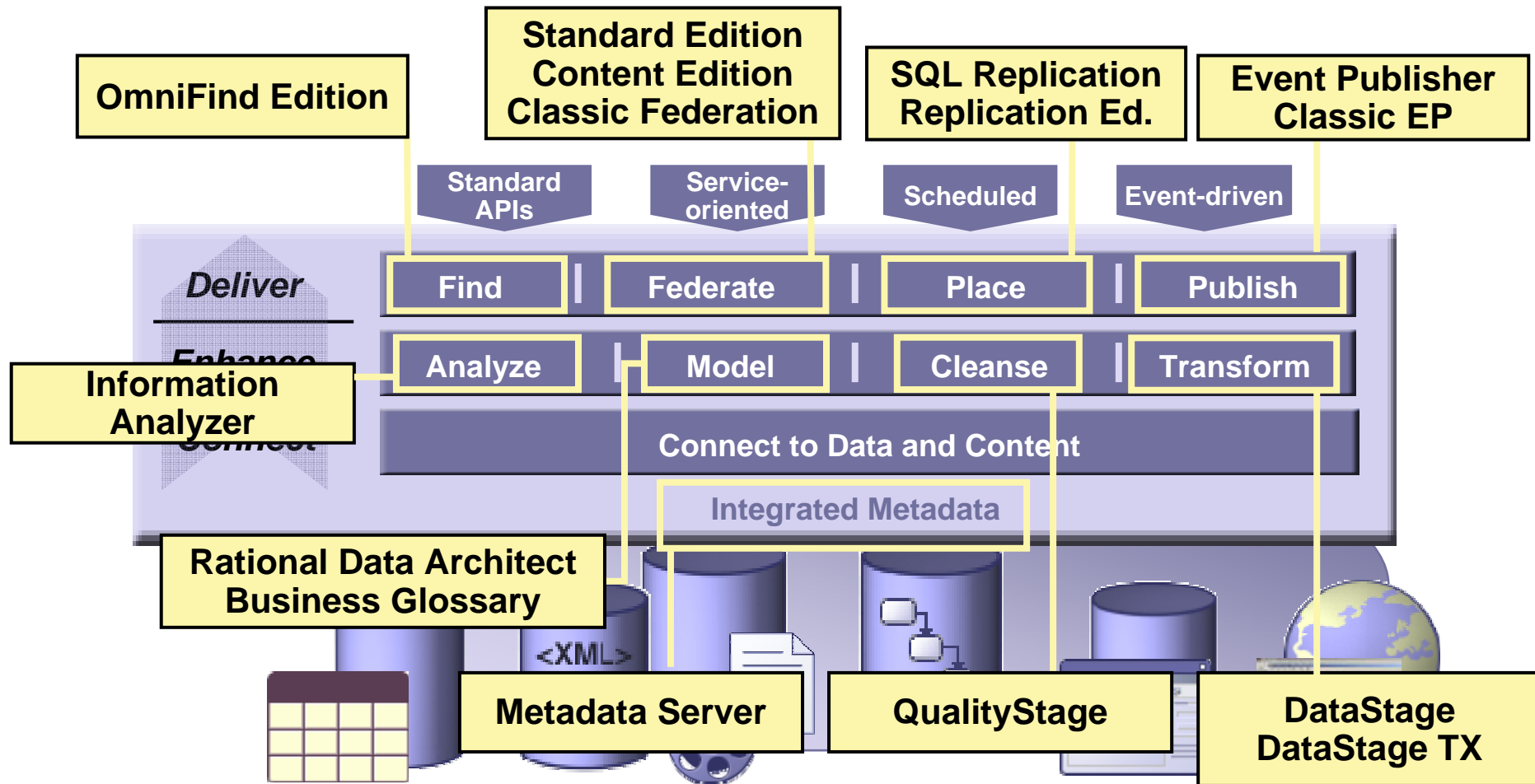


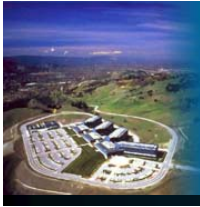
IBM Information Server Has It All ☺





Many Products to Choose From





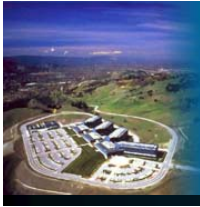
Customers want "I", not EAI, EII, ETL, ...

- Hard and dangerous choices
 - Too many products with too much functional overlap
 - Too hard to choose, have to choose too early
 - Too hard to switch among products
- Too hard to use to build effective solutions
 - Too many knobs, too much training to use well
 - Too stove-piped (hard or expensive to use in combination)
 - Too little easy life-cycle flow among products
- Too slow to succeed
 - Need services to survive
 - No way to kick-start the process
- Integration never ends
 - Integration usually starts with a project or two
 - Must be able to re-use infrastructure and expand gracefully



Back to the Future? Nonprocedural Data Access is What We (Still) Need

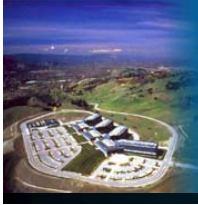
- Relational dbms provided nonprocedural access
 - Relational data model
 - Declarative query language
 - Relational calculus and algebra
 - Optimizer to map between them
 - Efficient algorithms for execution
- What has changed today?
 - Heterogeneous data in heterogeneous sources
 - Many if not most applications need data from multiple sources
- The tools we have today are far from relational simplicity
 - Multiple engines with different characteristics and interfaces
 - Integration specification depends on engine
 - Rarely declarative
 - Operations defined by engines' capabilities
 - No formal model of operations
 - No guidance on what engine(s) to use



Integration Nirvana

- Given: a set of *solution desiderata*
 - The information you want
 - By name: /abc/xqz/foo.txt
 - By type: pdf files
 - By properties: last_modified_date > "04/25/05"
 - By logical domain (business objects, e.g.): ...
 - The output you need
 - Form: Schema specification, ... constraints
 - Delivery: as an EJB, in a file, as a rowset, in a database table, Web Service, ...
 - The quality, quantity and quality of data required
 - Need: accuracy, availability, ...
 - Constraints: frequency, currency and completeness of data
 - Physical constraints
 - Amount of memory, processing power, etc
- Automatically generate the instructions to do the integration
- Result: "nonprocedural" data integration

NOT NECESSARILY A COMPLETE LIST!!!



Implications and Approaches

- Hypothesis:
 - Understanding the role of the desiderata on integration solutions is essential
- Theory
 - What desiderata can we formally model?
 - How can we incorporate desiderata in a theory of integration?
 - Can we construct an algebra for integration?
- Systems
 - How can we specify the desiderata?
 - Can we build an Advisor or Wizard for integration that will recommend which engine(s) to use?
 - Can we create a compiler that produces a script to invoke existing engines (integration blades)
 - Can we create a new integrated integration engine that will meet any combination of desiderata?
 - Depends on a well-understood integration algebra



Operationally, What Does It Mean?

- Integration will have converged
 - No visible difference between ETL, federation, replication
 - Process and information integration just work together seamlessly
 - Mapping will be done the same way for all uses
 - Metadata will be captured once and used everywhere
- Most of integration development, deployment done automatically
 - Auto-discovery of metadata, data, configuration
 - Automatic/assisted design of most integration
 - Logical integration design compiled into integration machinery
 - Auto-choice of combination of engines to satisfy QoS requirements
 - Auto-monitoring, dynamic tuning
- New applications easy to develop over legacy and new data
- Legacy applications run unperturbed on newly available data



The Big I: Convergence and Unification?

