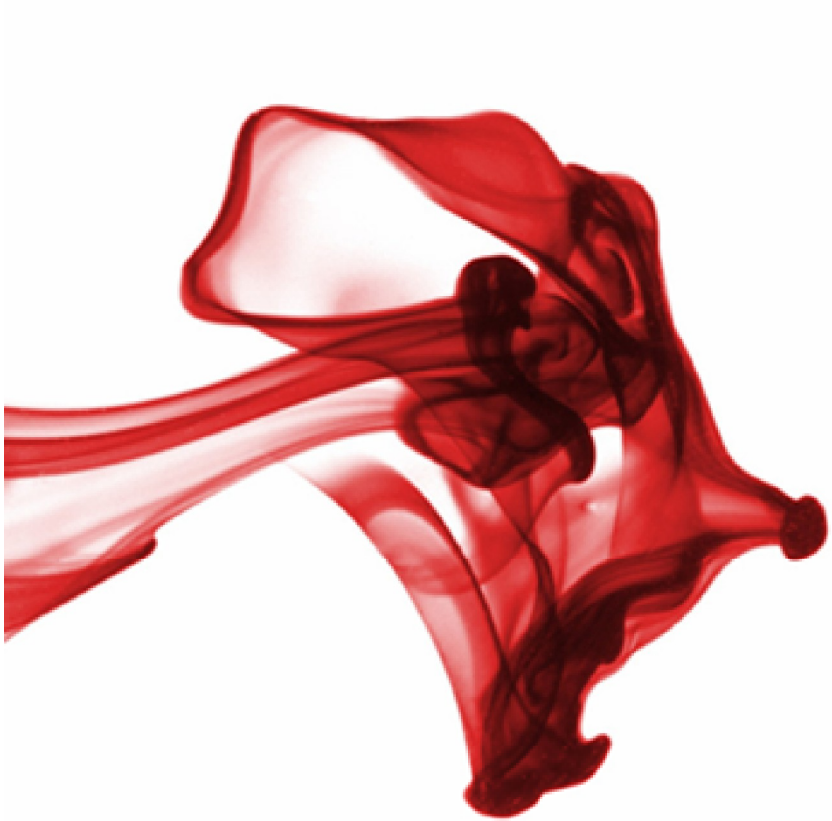


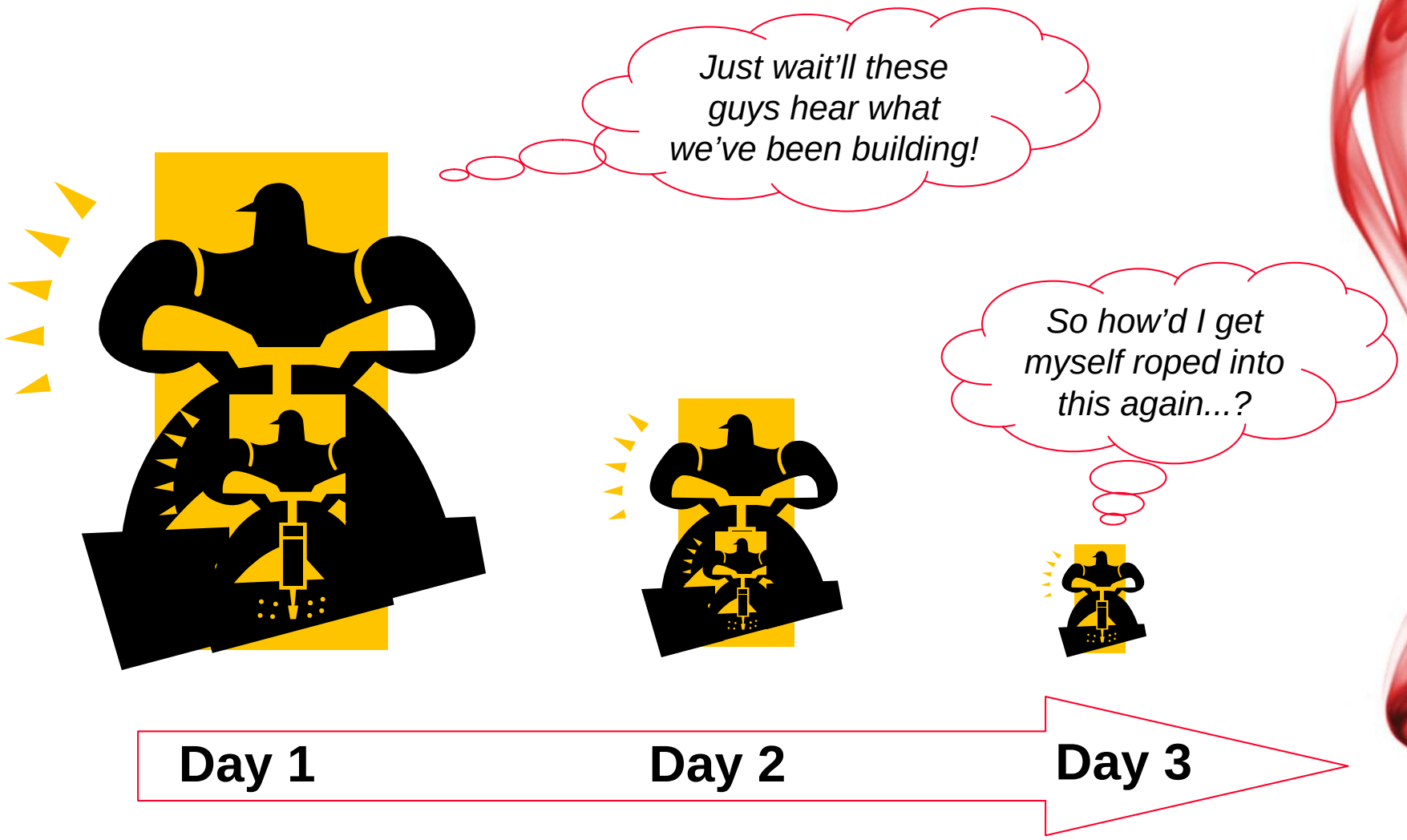
SOA What?

Michael Carey

BEA Systems, Inc.
(*mcarey@bea.com)*

bea
Think liquid.

I’d forgotten about HPTS(t)...



 |

Roadmap

- Data Then and Now
- Data in a SOA World
- So Many Choices, So Little Time
- A Call to Arms

bea Think liquid.

Roadmap

- Data Then and Now
- Data in a SOA World
- So Many Choices, So Little Time
- A Call to Arms

bea Think liquid.

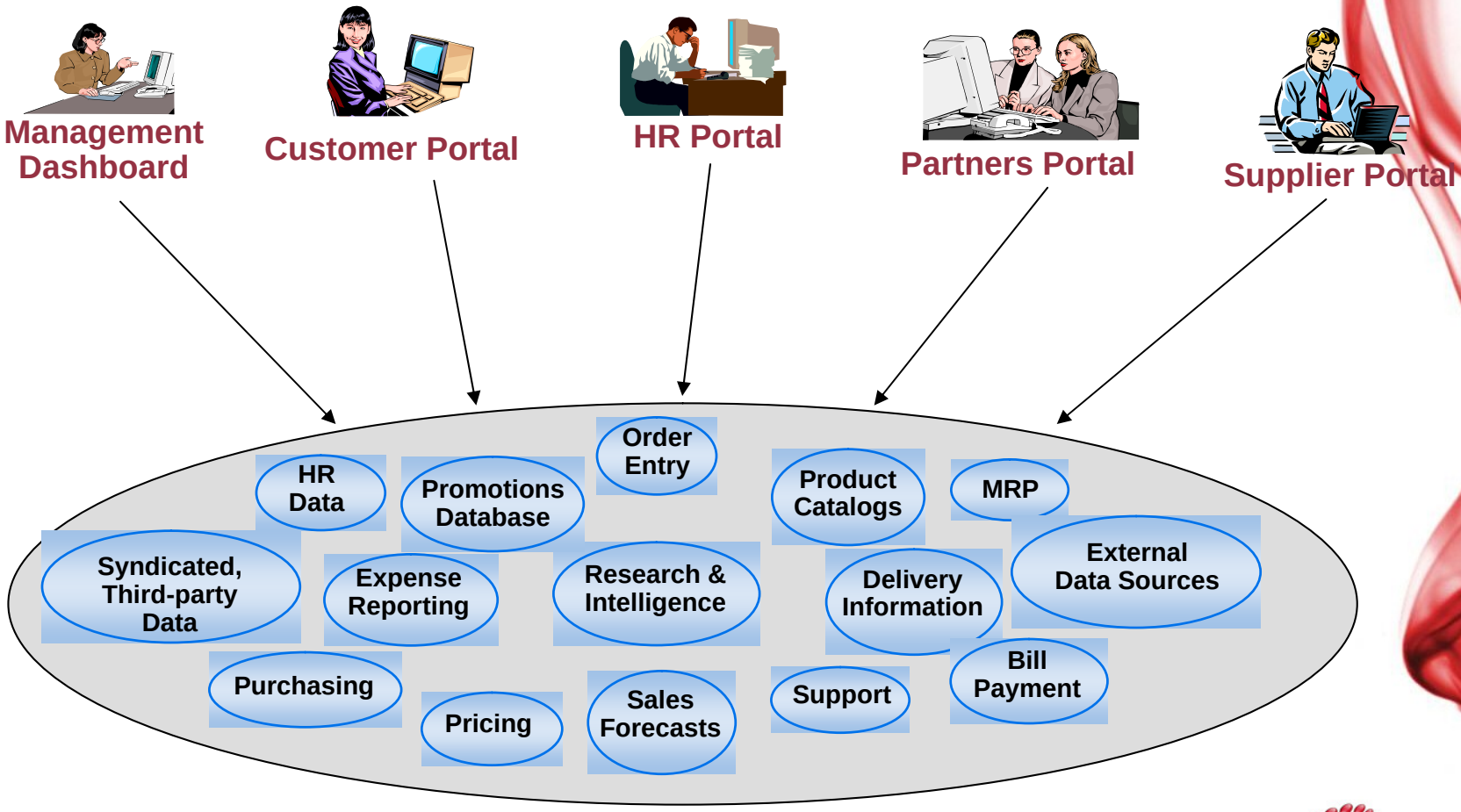
Once Upon a Time in a Galaxy Far, Far Away

- Databases were methodically designed for applications
  - E-R modeling based on “business object” analysis
  - Map E’s & R’s to tables, continue design from there
- Databases were relational and lived in glass houses
  - Simple model (tables) and language (SQL)
  - Major productivity gains from declarative programming and data independence
- Application development was pretty straightforward
  - Client/server application architecture (SQL, SPs)
  - Embedded SQL, ODBC/JDBC, EJBs, ORM tools
- Life was simple, life was good – sigh…

Data Here, Data There, Now I've Got Data Everywhere

- Perhaps relational databases made things too easy?
  - Departmental vs. inter-galactic centralized databases
- Databases come in many flavors
  - Relational: Oracle, DB2(s), SQL Server, MySQL, …
  - Hangers-on: IMS, IDMS, VSAM, …
- Not all important data is SQL-accessible (or even relational)
  - Packaged apps: SAP, PeopleSoft, Siebel, Oracle, …
  - SaaS apps: SalesForce, Workday, Rightnow, …
  - Custom "homegrown" apps
  - Files of various shapes and sizes
  - Data in Mike Stonebraker's non-one-size-fits-all systems (☺)
  - And the list goes on…

The Modern Enterprise (as Depicted by Marketing ☺)



IT Landscape

bea Think liquid.

This is Painful for Application Developers (Akin to Pre-Relational Era?)

- No one “single view of X” for any X
  - What data do I have about X?
  - How do I stitch together the info I need about X?
  - What else is X related to?
- Heterogeneity in multiple dimensions
  - *Model heterogeneity:* disparate data models and/or data formats
  - *API heterogeneity:* disparate data access, service call, and update APIs
  - *Schema heterogeneity:* disparate representations for the same info
- No reuse of developed artifacts
  - Different access criteria & returned data → different stitching schemes
  - How would anyone even begin to find such views/artifacts (*w/o* a model)?

Never Fear – I've Read the Trade Rags... It's SOA To The Rescue!

- Service-Oriented Architecture (SOA)
  - Loosely-coupled interfaces (e.g., Web service contracts)
  - Each subsystem is a component with a service API
  - Create new assets by integrating & composing your existing assets!
- We're closer to dealing with heterogeneity
  - Services all have XML Web service foundations
  - Custom logic is hidden (e.g., data access and/or integration)
- Fine …. but what about my data…?
  - What are my business entities and how are they interrelated?
  - How can I find them and/or share them, and what can they do?
  - And where did my nice, declarative, data-centric world go...?

→SOA ***What***?

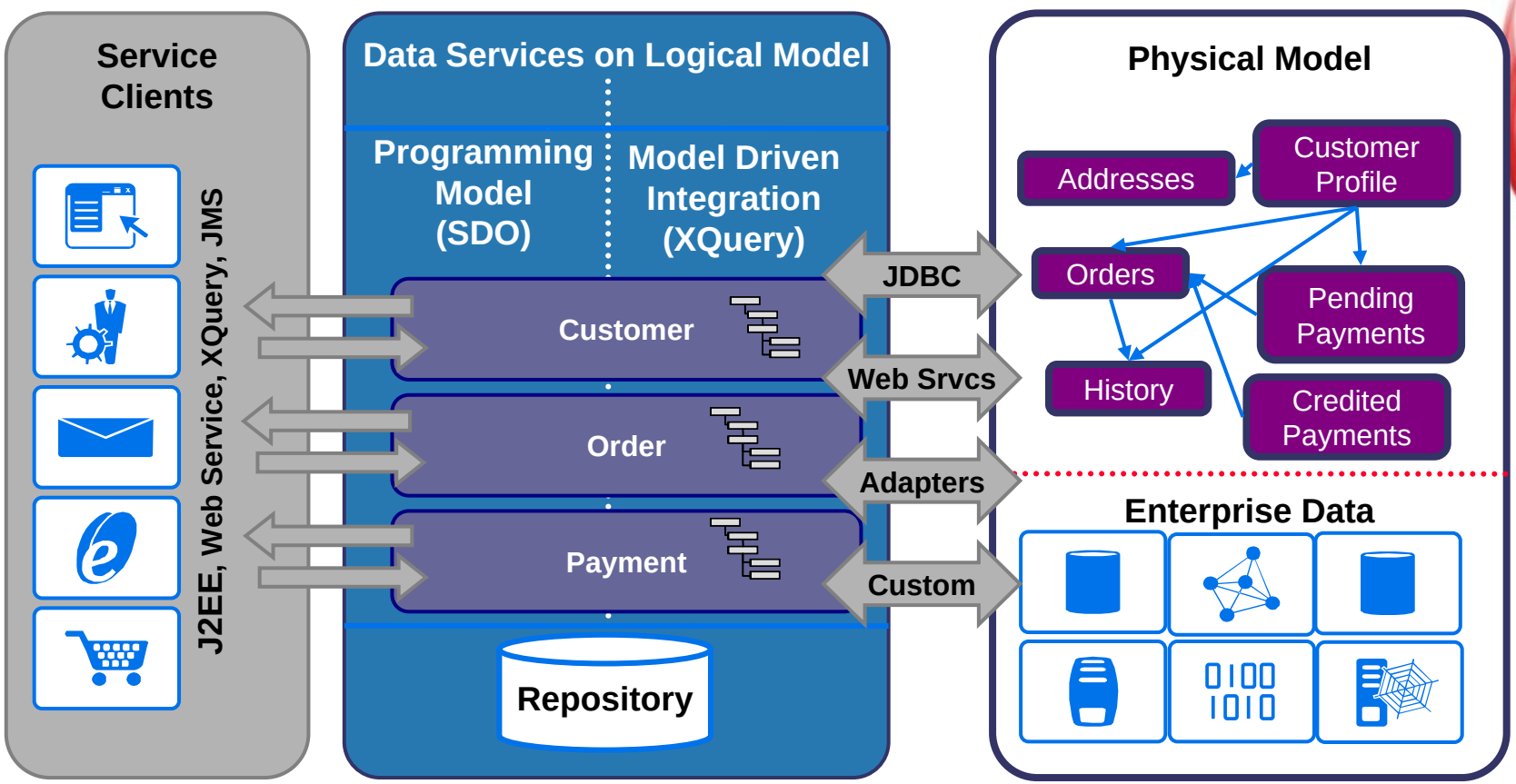
bea Think liquid.

Roadmap

- Data Then and Now
- Data in a SOA World
- So Many Choices, So Little Time
- A Call to Arms

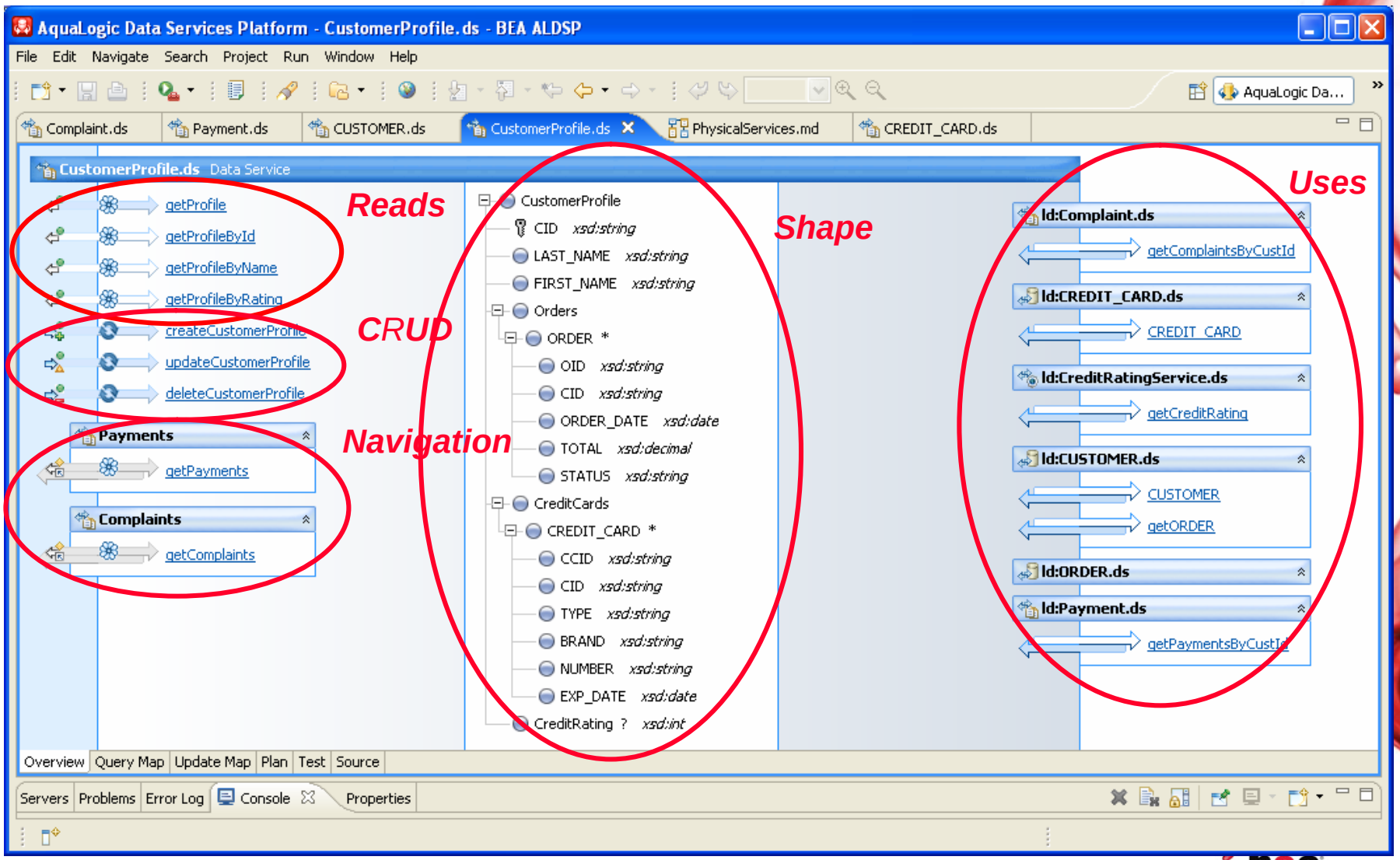
bea Think liquid.

Data Services (*a la* BEA AquaLogic DSP)



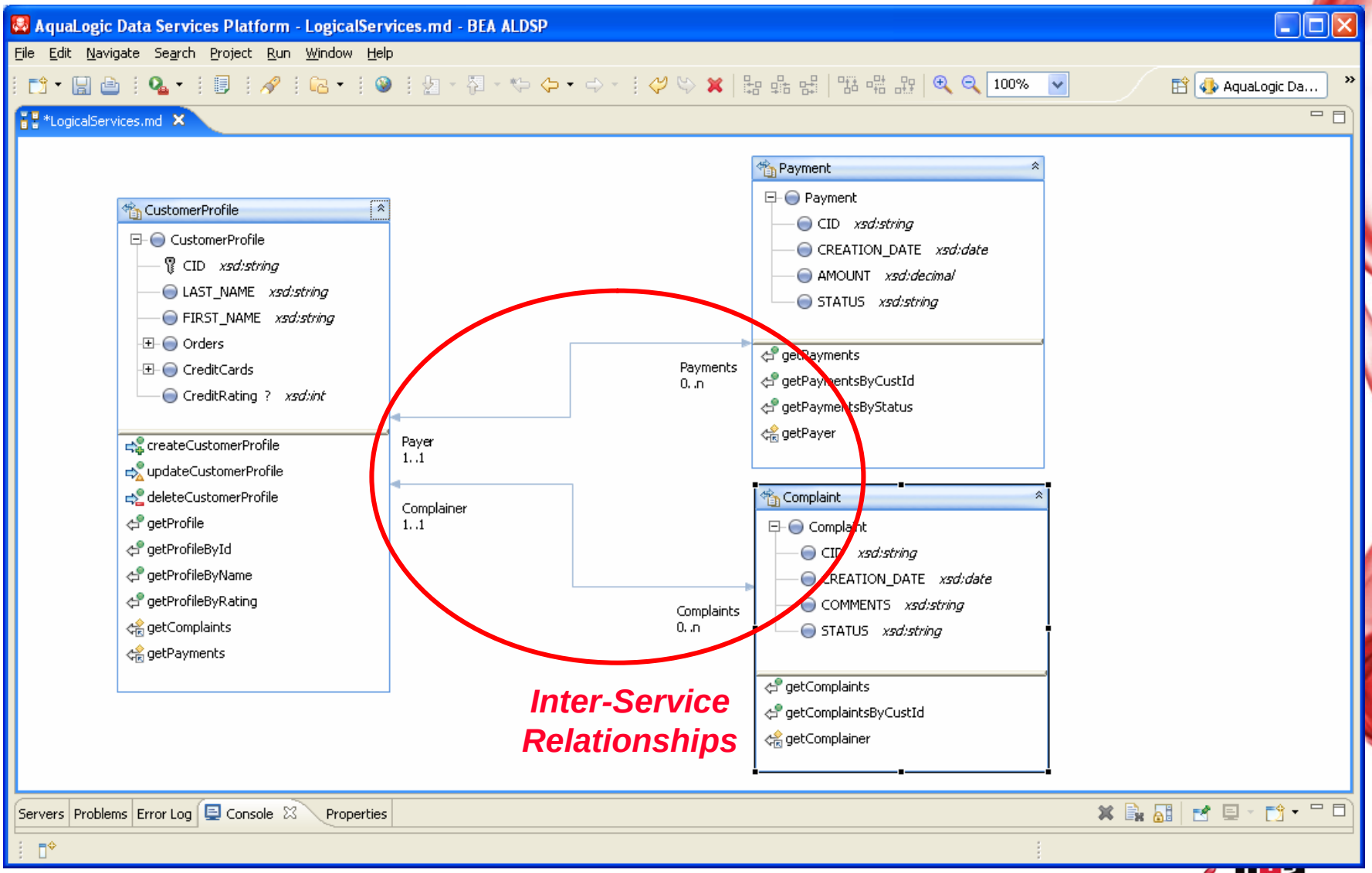
- Data services group operations related to (coarse-grained) **business entities**
- Logical models capture data access and integration complexity **one time**
- **Uniform** data model, programming model, and API for all enterprise data
- Service **clients** may be portals, business processes, composite applications,...

ALDSP Data Services – Overview

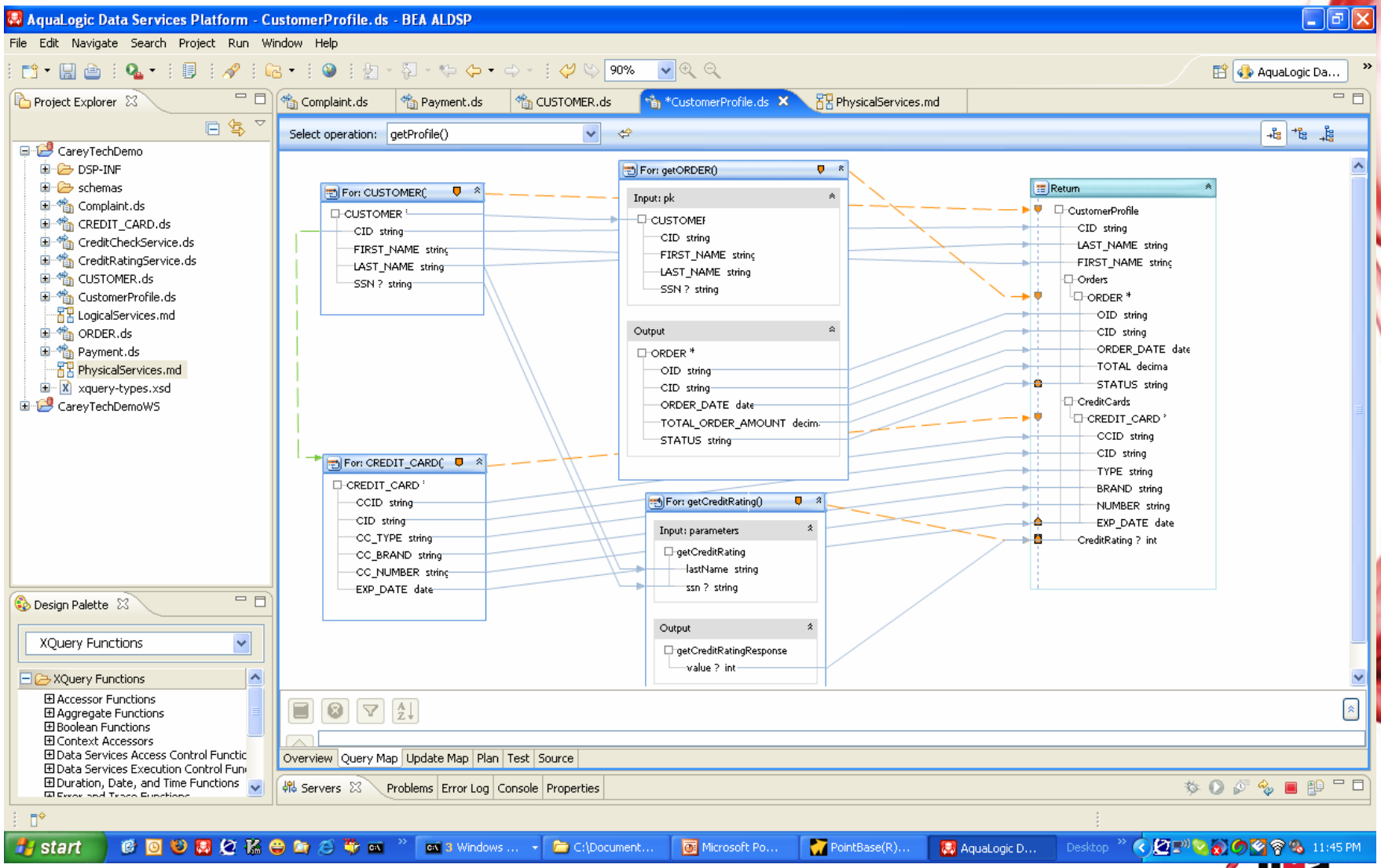


bea Think liquid.

Interrelated Data Services

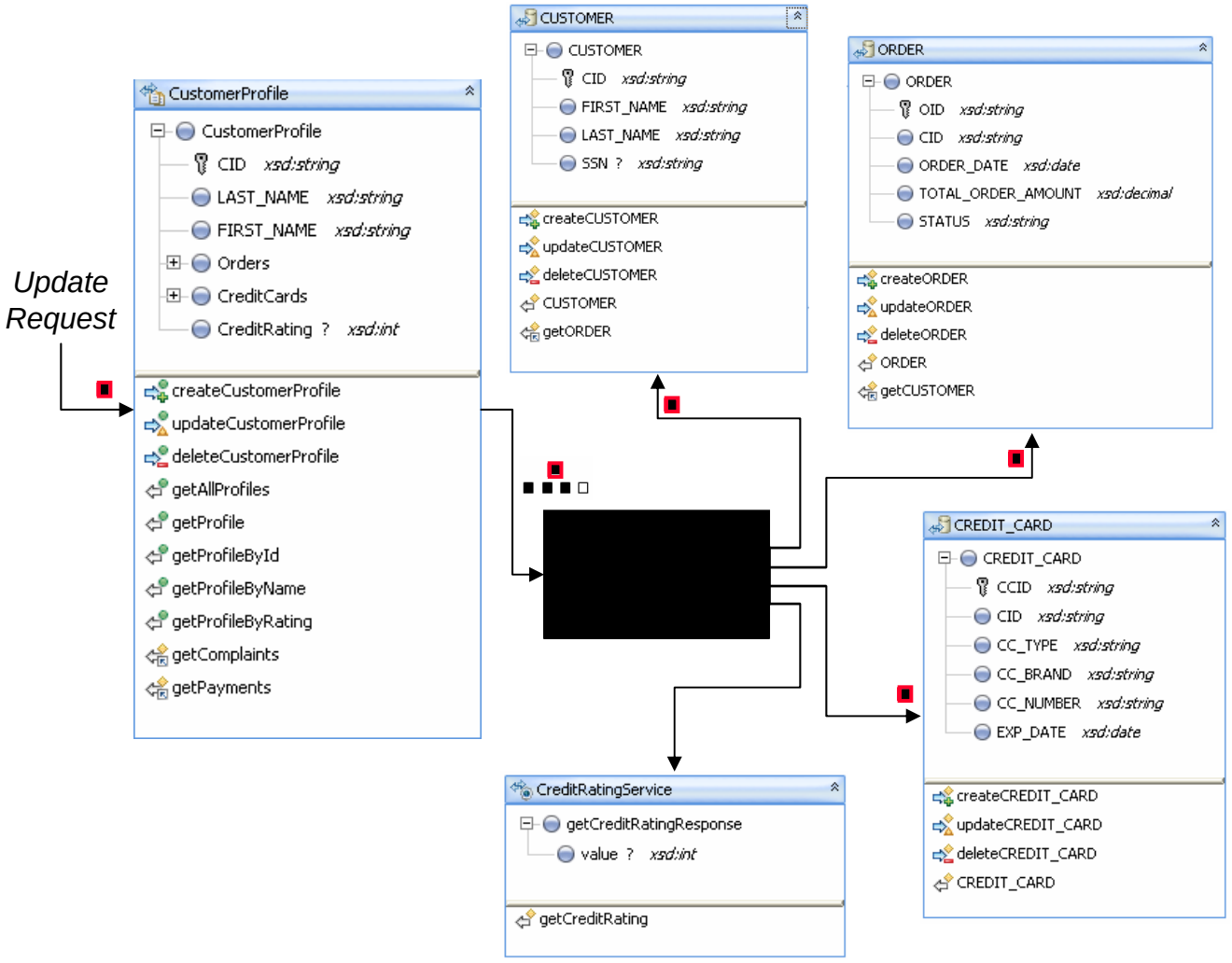


Graphical Service Authoring



Think liquid.

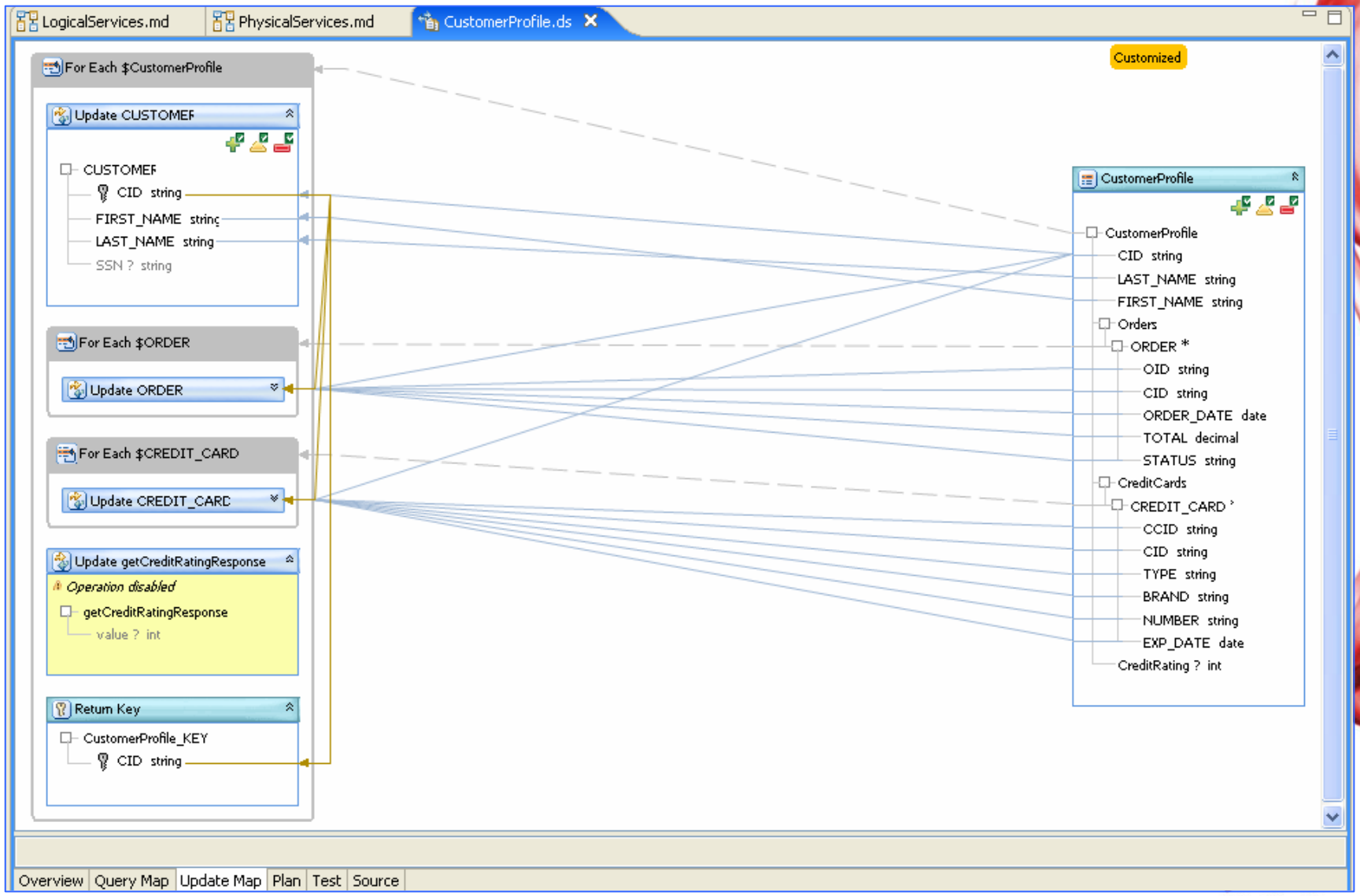
Inferring Update Services



Update Framework

- Automatic change decomposition
- Graphical update map editor
- SQL generation for RDBMSs, including optimistic concurrency
- XA and non-XA data sources
- Reliable non-XA error logging
- User-defined CRUD operations, in Java or XQSE for business validation, update replacement logic, etc.
- Next: compensation (via declaration of undo/did-I-do actions)

Graphical Update Map Editor



Think liquid.

Roadmap

- Data Then and Now
- Data in a SOA World
- So Many Choices, So Little Time
- A Call to Arms

bea Think liquid.

Various Commercial Technologies are Vying for Enterprise IT's Dollars

- Build a virtual (federated) database using "EII"
  - Uniform and unified querying over current data on X
- Build an operational data store or a master data hub
  - Place to replicate cleansed, consolidated data about X
- Build a data warehouse or a data mart
  - Great place for doing analytics and mining of X's history
- Use ETL and data matching/cleansing tools
  - These are a means to some of the above ends
- All these technologies have a place, but they're missing a big opportunity to relieve developer pain…

Let's Consider a Small Example

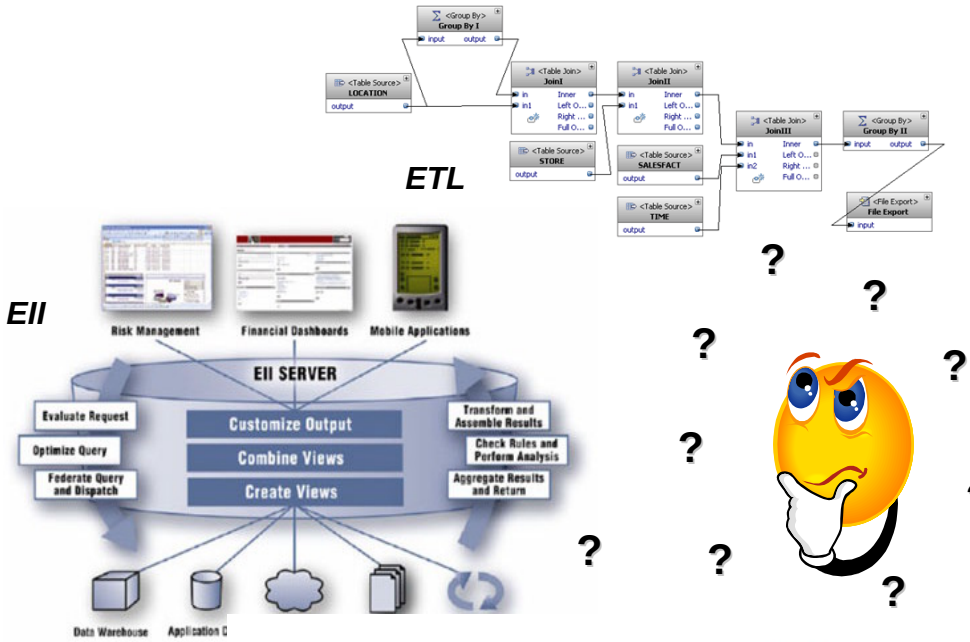
- Customer data in 4 sources – 2 RDBs and 2 Web services
  - A customer contact database
  - A CRM system (e.g., Siebel)
  - An order management system (e.g., SAP)
  - A service case management database
- Could use EII to create a single view of Customer
  - Encapsulate the logic to transform and relate source data
  - Provide a Customer view that can be queried/reused by apps
  - Provide a basis for higher-level (a.k.a. external) views
  - Enable query optimization – e.g., data source elimination, predicate pushdown, and so on

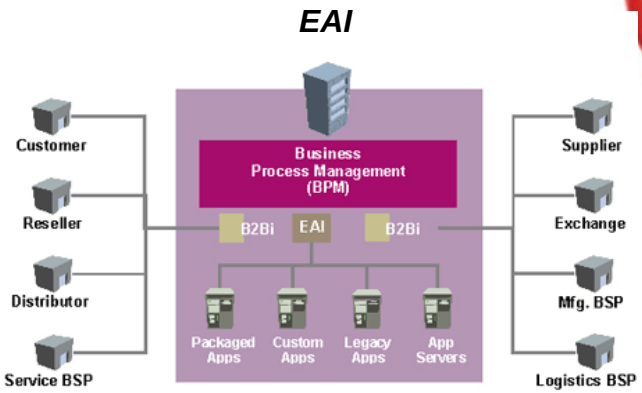
Let's Consider a Small Example *(cont.)*

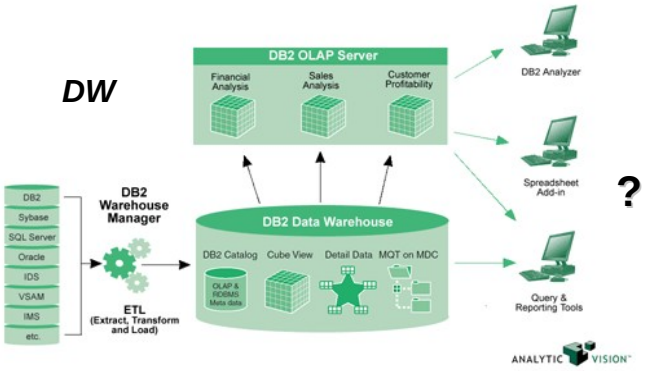
- Could use EAI to update our single view of Customer
  - Write a workflow to coordinate the cross-system changes
  - The workflow can also help deal with non-XA sources
  - Many real integration scenarios will *require* workflows
- So what's wrong with this EII + EAI answer?
  - The EII view says how to fetch, transform, and match data from the four sources
  - The EAI workflow says how to apply updates to the four sources, re-specifying the sources, transforms, matching logic, and so on
  - And, if we were also doing ETL into a warehouse, ...
  - Integration architects today have to repeat themselves (using multiple programming paradigms and tools)!

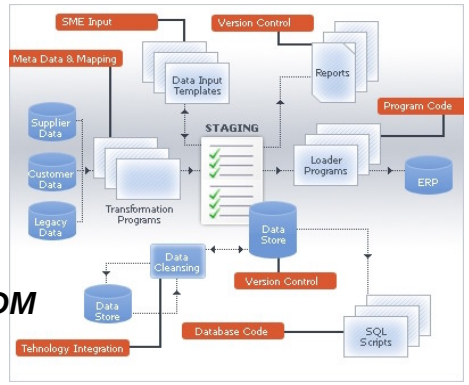
Q: Data problem or process problem or...?

EII ... EAI ... ETL → **III...!**
(Pronounced *"Aye Yi Yi...!"* ☺)









bea Think liquid.

What's Really Needed Here: *EI*

- Integration information captured *just once*
  - One model to learn, one metadata repository to populate
- Declarative approach to queries *and* updates *and* data transformation *and* ...
  - Want all the benefits of EII for queries
  - Should also be able to (help) automate update processes
  - Should be applicable to ETL transforms as well
  - And so on...
- Eliminate the EAI/EII/.../EIEIO distinction
  - Don't make users choose between these "approaches"
- Interesting challenges in this area include
  - Integrating services (i.e., applications) as well as data
  - Choice of overall model, required metadata, capturing of semantics (for mappings and services), …

Roadmap

- Data Then and Now
- Data in a SOA World
- So Many Choices, So Little Time
- A Call to Arms

bea Think liquid.

SOA There!

- Our customers are facing an EI problem in SOA land
  - Enterprises are moving towards services and SOA
  - Data is often service-fronted and only service-accessible
  - If we can't deal with this, we (data folks) can't help them anymore!
- The IT world needs us to help lead them to a place where...
  - Apps can easily find, consume, and produce services
  - New apps can be built via composition / orchestration
    - "Megaprogramming" (Ceri, Wegner, and Wiederhold)
  - Data (and data modeling) doesn't fall by the wayside
  - Declarative programming doesn't fall by the wayside
- This is a call to arms for E** and SOA researchers, developers, and vendors...

Q&A

- Questions?

bea Think liquid.

Application Development in that Galaxy

Department

dno	name
10	Toy
20	Shoe

Employee

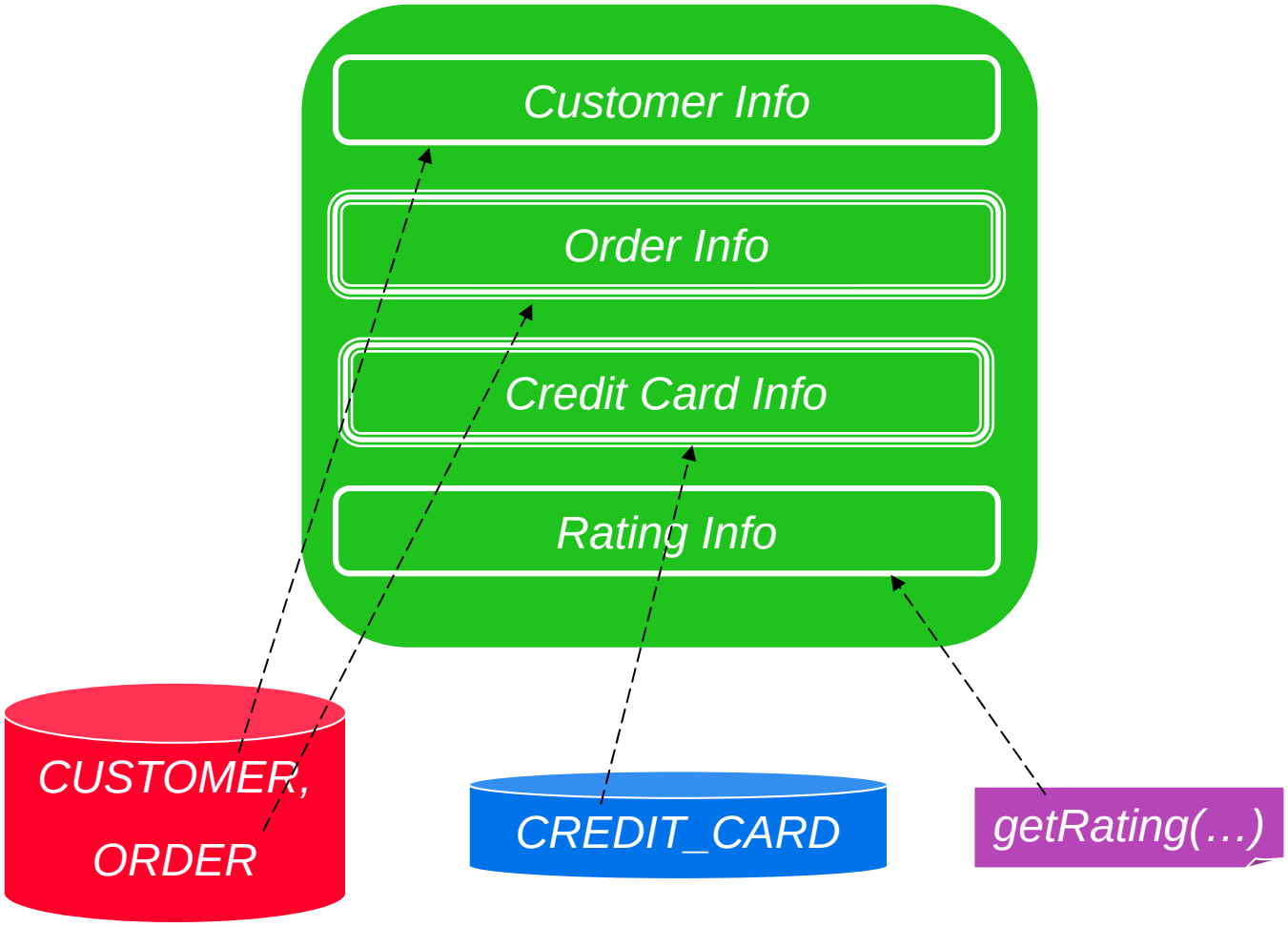
eno	name	salary	dept
1	Lou	10000000	10
7	Laura	150000	20
22	Mike	80000	20

```
...
stmt = dbconn.prepareStatement (
  "select E.name, E.salary, D.no
   from Employee E, Department D
   where E.salary < 100000
      and D.name = ?
      and E.dept = D.dno"
);
...
```

Declarative Integration via XML & XQuery

Requirements		Standards
A standard for data format and data interchange	→	XML
A standard for describing and modeling data	→	XML Schema
A standard for interfacing into applications	→	Web Services
A standard for querying both relational and non-relational data	→	XQuery
A standard Java programming model (read + write)	→	SDO (Service Data Objects)
A standard for publishing consumable data services	→	Web Services

Ex: Customer Profile Data Service



Service Data Objects (SDO)

```
//Get SDO
CustomerDoc custSDO =
    CustomerDS.getCustomerById("007");
```

Original SDO

```
<CustDataGraph>
<cus:CUSTOMER xmlns:cus="ld:LiquidDataApp/CUSTOMER">
<CUSTOMER_ID>007</CUSTOMER_ID>
<CUST_NAME>Michael</CUST_NAME>
<EMAIL_ADDRESS>mikejcarey@aol.com</EMAIL_ADDRESS>
<TELEPHONE_NUMBER>408-570-8599</TELEPHONE_NUMBER>
</cus:CUSTOMER>
</CustDataGraph>
```

```
// Make changes to SDO
custSDO.setCustName("Mike");
custSDO.setEmail("mcarey@bea.com")
```

SDO w/ Changes

```
<CustDataGraph>
<cus:CUSTOMER xmlns:cus="ld:LiquidDataApp/CUSTOMER">
<CUSTOMER_ID>007</CUSTOMER_ID>
<CUST_NAME>Mike</CUST_NAME>
<EMAIL_ADDRESS>mcarey@bea.com</EMAIL_ADDRESS>
<TELEPHONE_NUMBER>408-570-8599</TELEPHONE_NUMBER>
</cus:CUSTOMER>
<ChangeSummary>
<CUSTOMER com:ref="/CUSTOMER">
<CUST_NAME>Michael</CUST_NAME>
<EMAIL_ADDRESS>mikejcarey@aol.com</EMAIL_ADDRESS>
</CUSTOMER>
</ChangeSummary>
</CustDataGraph>
```

```
//Submit SDO
CustomerDS.submit(custSDO);
```


bea Think liquid.