

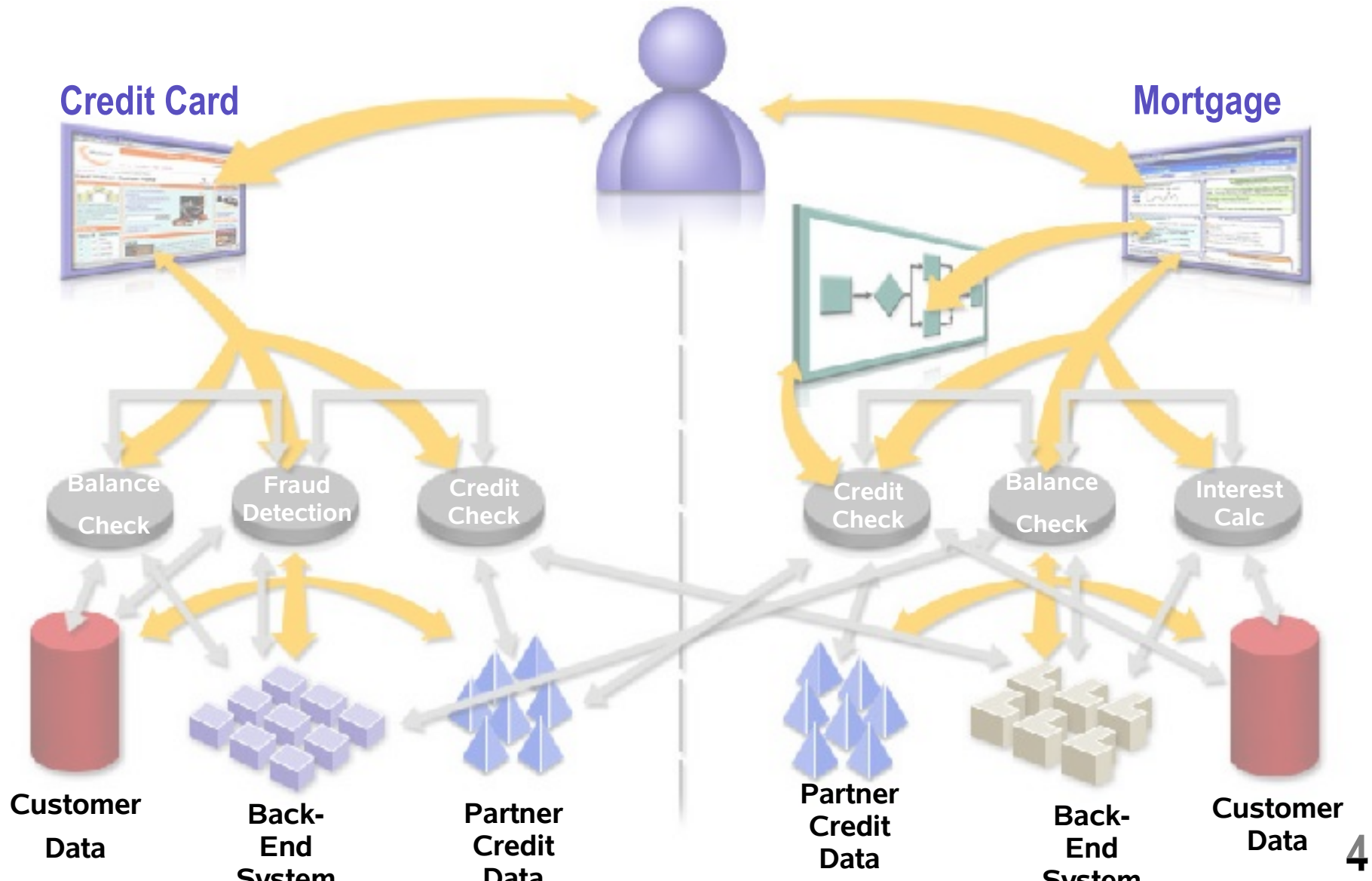
SOA Basics

Topics

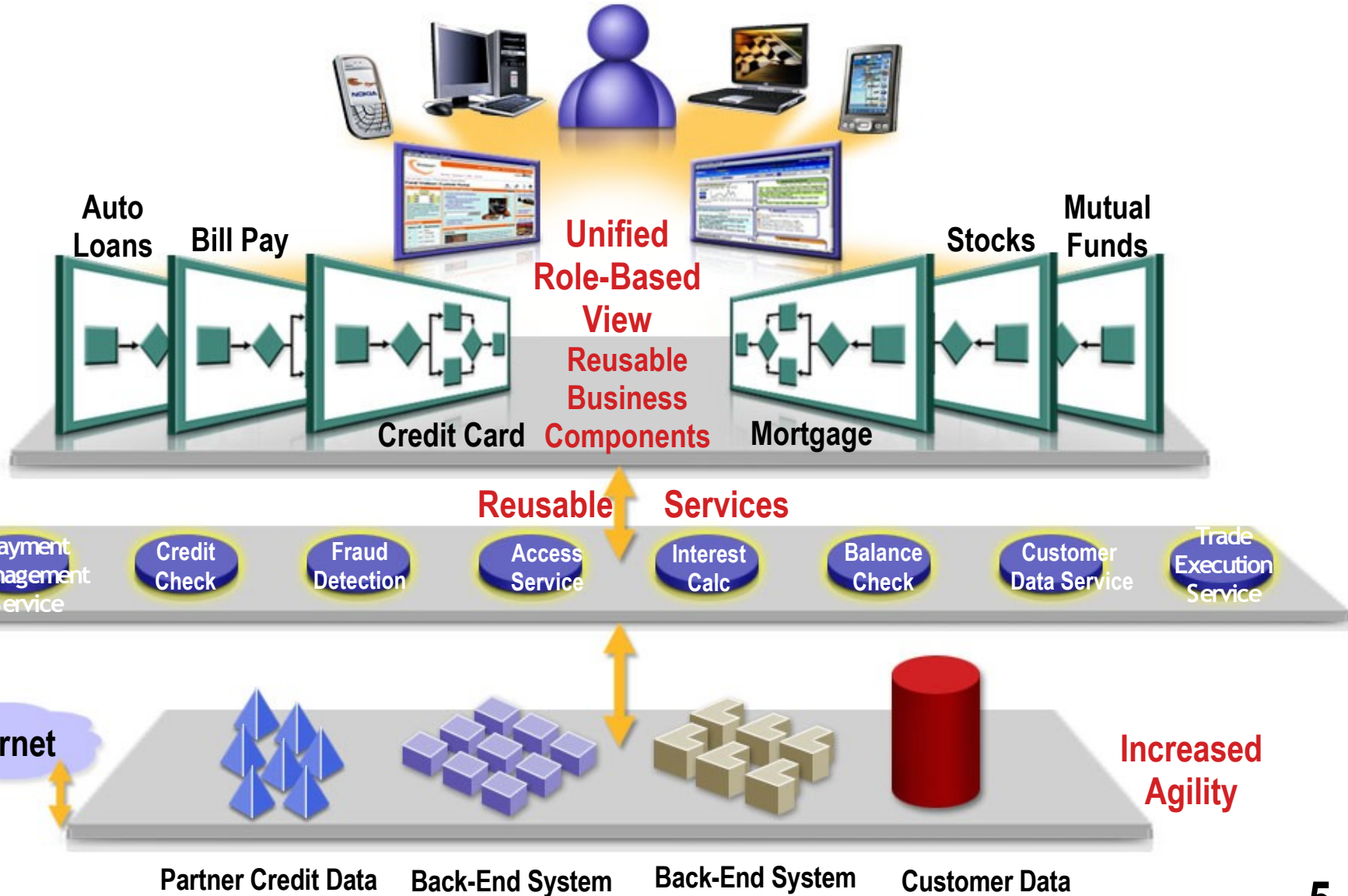
- Quick introduction to SOA
- Composite applications
- Services and SOA

Quick Introduction to SOA

Pre-SOA Scenario



SOA-Enabled Scenario



SOA Layers

- Shared Network-based **Layered** Services

Access Layer

Process (Orchestration) Layer

Service Layer

Resource Layer

Benefits of SOA

- Flexible (Agile) IT
 - > Adaptable to changing business needs
- Faster time to market
 - > Reuse existing code, minimize new development
- Business and process-driven
 - > New business opportunities
- Greater ROI
 - > Leverage existing IT asset

Service Oriented Architecture (SOA)

- An architectural principle for structuring systems into coarse-grained *services*
- Technology-neutral best practice
- Emphasizes the loose coupling of services
- New services are created from existing ones
in a synergistic fashion
- Strong service definitions are critical
- Services can be re-composed when business requirements change

Composite Applications

Applications

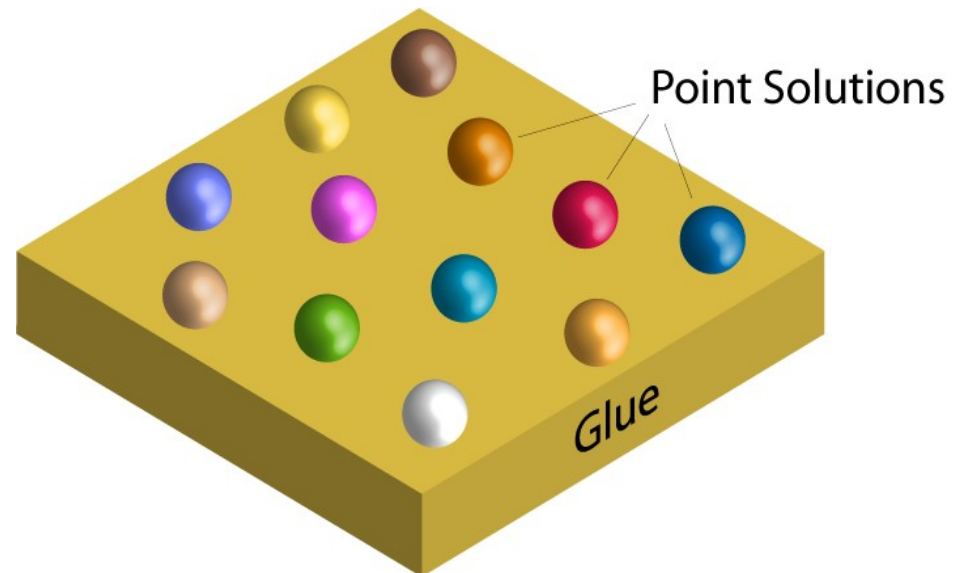
- Developers need to build end-to-end applications
 - > Front-end user interfaces
 - > Middle-tier business logic
 - > Back-end resources
- With the right approach, developers can...
 - > Reuse existing parts
 - > Build new parts
 - > Glue old and new parts together
- With the wrong approach, developers must...

Applications

- Real-world applications are...
 - > ...not Web applications
 - > ...not Java EE applications
 - > ...not Swing forms
 - > ...not Web services
 - > ...not BPEL processes
 - > ...not SOA
 - > ...not JBI
 - > ...not RDBMSs
 - > ...not (your favorite technology)
- Real-world applications use many or all of

Traditional Application Development

- Point technologies, products, and APIs
 - > For example: EJB, Spring, Hibernate, JSF, Servlets, Struts, etc.
- Lots of glue written by developers
 - > Requires a great deal of expertise & time
 - > Inflexible



Composite Applications

- A way to compose applications from reusable parts
- Composite applications employ SOA principles
 - > Features exposed as Web services
 - > Standards-based interaction between services
 - > Are themselves composable

Services and SOA

What Are Services?

- Black-box components with well-defined interfaces
 - > Performs some arbitrary function
 - > Can be implemented in myriad ways
- Accessed using XML message exchanges
 - > Using well-known message exchange patterns (MEPs)
- Metadata in the form of WSDL describes...
 - > Abstract interfaces
 - > Concrete endpoints

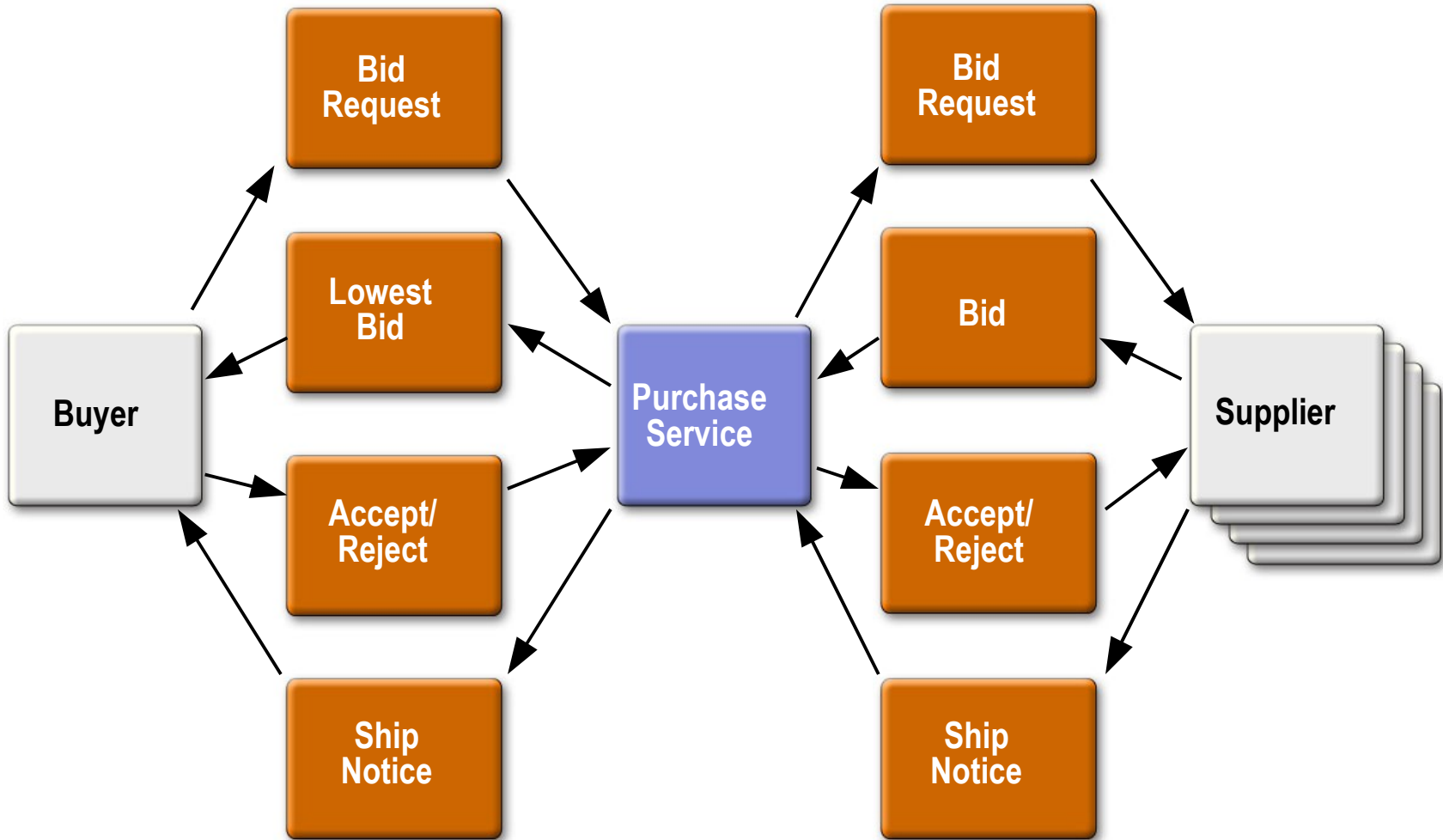
What Can Services Do?

- Perform business logic
- Transform data
- Route messages
- Query databases
- Apply business policy
- Handle business exceptions
- Prepare information for use by a user interface
- Orchestrate conversations between multiple services

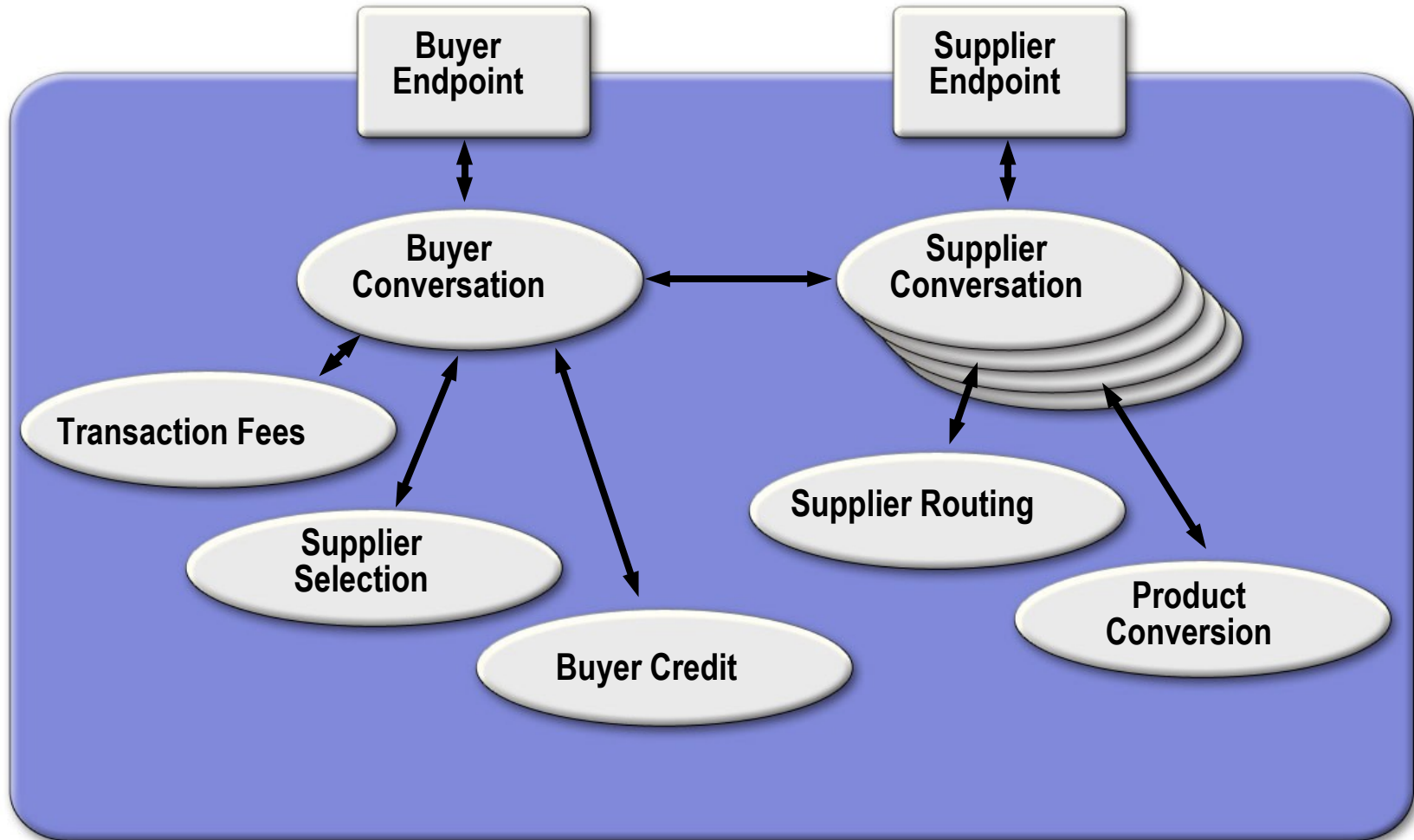
How Are Services Implemented?

- Enterprise JavaBeans™ (EJB™) technology
- BPEL
- XSLT
- SQL
- Business rules
- Mainframe transaction
- EDI transform
- Humans (yes, really!)
- ...

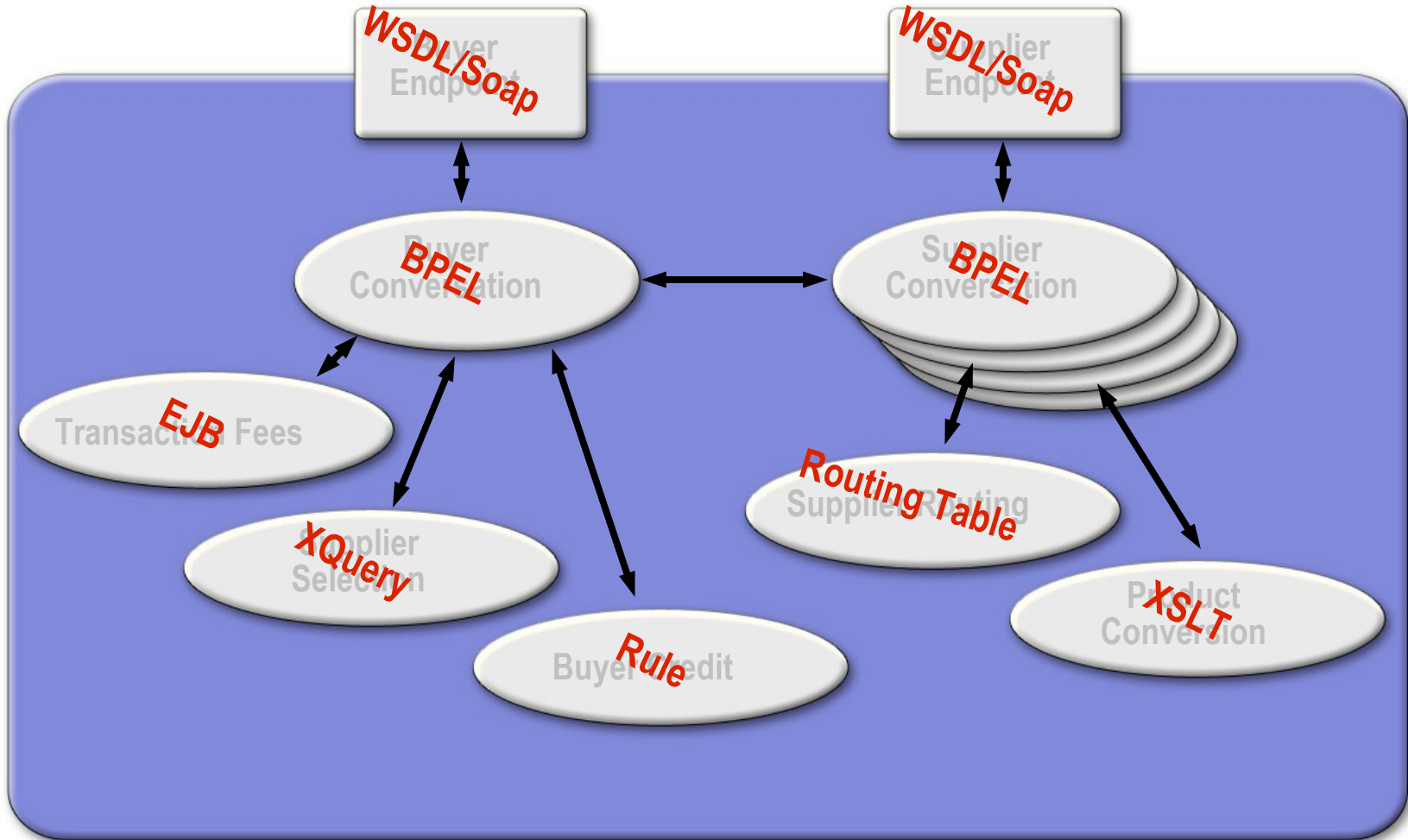
Example: Purchase Service



Purchase Service Functions



Purchase Service Functions



SOA Basics