# 1. Key Components of Distributed Applications

- consume data from distributed data sources

Designed for:

* Scalability
* Low latency
* Availability
* Reliability
* Security and privacy

## Logical Layers of Distributed Applications

* Data Layer
* Business Layer
* User Interface Layer
* Service Layer

### Data can be represented in different models:

* Relational (databases, tables and columns)
* Hierarchical (XML, JSON)
* Object oriented (entities in code)

### There are many types of data store:

* Relational database
* File-systems and distributed file-systems
* Distributed caches
* No SQL databases
* Cloud storage
* In-memory stores

### .Net Data Technologies

* System.IO
* ADO.NET
* Entity Framework
* In-Memory cache (system.web)
* Windows AppFabric cache, Windows Azure caching

Http can also be used for accessing data

* OData services with WCF Data Services or ASP.NET WebAPI
* Windows Azure Storage

Declarative data queries in C# using LINQ

### SOAP and Http-based services

SOAP based

* Based on SOAP, an XML based format
* Use a Remote Procedure Call (RPC) approach
* Interoperable over HTTP, UDP, SMTP and AMQP

Http-based

* Support multiple content types (XML, Text, Images)]
* Uses a resource based approach
* Http is underlying protocol of the world wide web

### Cloud computing

* Infrastructure as a Service (IaaS) – virtual machines
* Platform as a Service (PaaS) ready to use platform that provide application hosting that can be cloned and scaled automatically
* Software as a Service (SaaS) – ready to use on demand software

### Windows Azure Cloud Services

* PaaS sollutions for your application:
  + Stateless
  + Provision ready
  + Scalable
* Cloud services host applicationson role:
  + Web role – IIS based applications
  + Worker role – host for any type of process, services, background processing etc

### Windows Azure Application Components

* Storage Services
* Storage Bus
* Access Control Service
* Distributed Cache
* Content Delivery Network
* SQL Databases as a Service

### Windows Azure Storage

* **Blob** **storage** – non-structured collection of objects accessed by a resource identifier
* **Table storage** – semi-structured collection of objects that can have fields but not relations between objects
* **Queue Storage** – provides a persistent messaging queue