CISC 322

Software Architecture

Lecture 04:

Non Functional Requirements (NFR) – Quality Attributes

Emad Shihab

Adapted from Ahmed E. Hassan and Ian Gorton

Last Class - Recap

- Lot of ambiguity within stakeholders
- Focus on the needs, not wants
- Specifications used to bridge gap between stakeholder demands and software system
- Use system perspective diagram to isolate system from users and interfaces

What are Quality Attributes

- Often know as –ilities
  - Reliability
  - Availability
  - Portability
  - Scalability
  - Performance

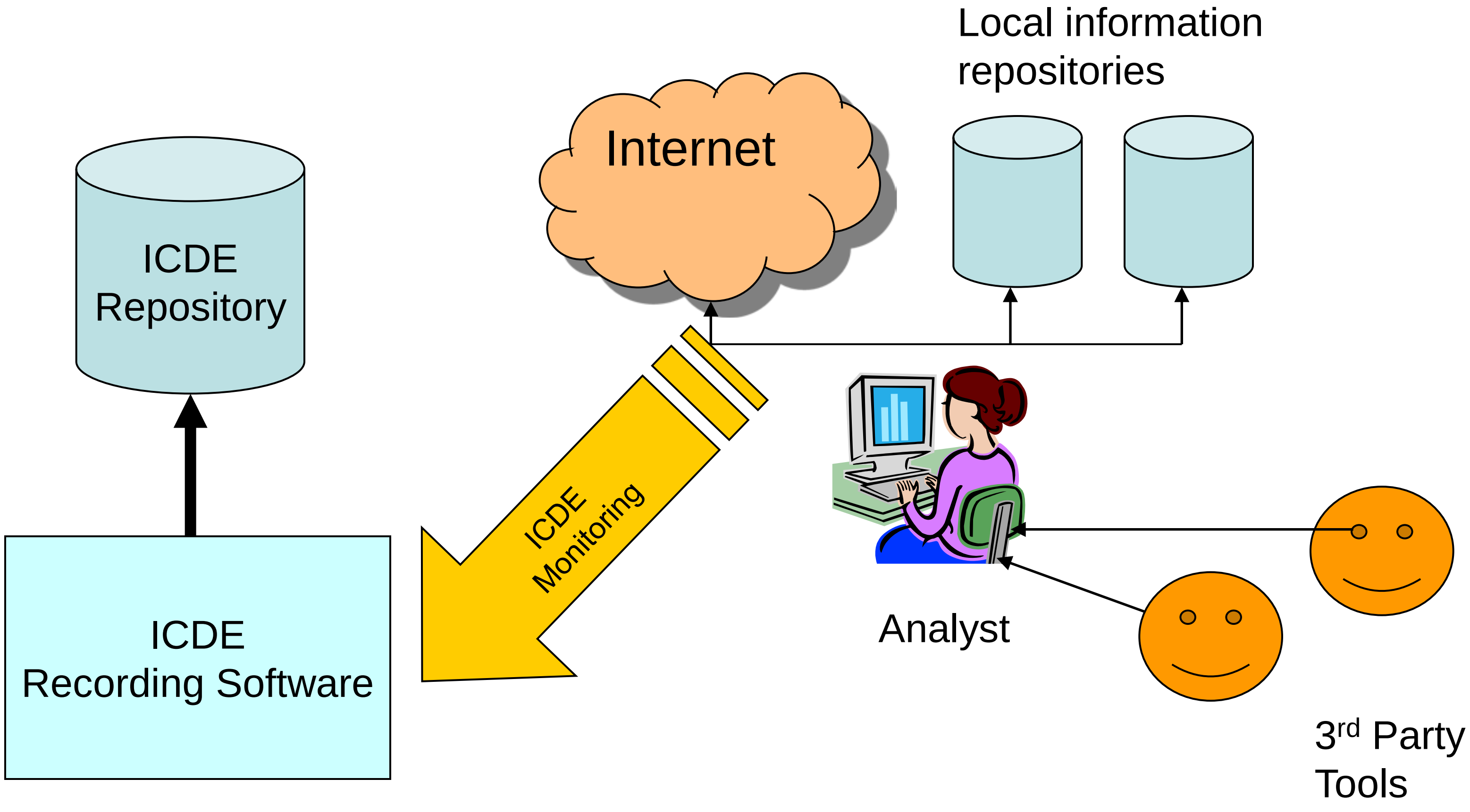
ICDE System

- Information Capture and Dissemination Environment (ICDE) is a software system for providing intelligent assistance to
  - financial analysts
  - scientific researchers
  - intelligence analysts
  - analysts in other domains

ICDE

- Automatically captures and stores data of actions performed by a user
- Google search:
  - Record query
  - List of returned pages
- Data can be later used by 3rd parties to offer intelligent help

ICDE Schematic



ICDE v2.0 Business Goals

Business Goal	Supporting Technical Objective	
Encourage third party tool developers	Simple and reliable programmatic access to data store for third party tools	Integration
	Heterogeneous (i.e. non-Windows) platform support for running third party tools	Portability
	Allow third party tools to communicate with ICDE users from a remote machine	Reliability
Promote the ICDE concept to users	Scale the data collection and data store components to support up to 150 users at a single site	Scalability
	Low-cost deployment for each ICDE user workstation	Scalability

Quality Attribute Specification

- Architects are often told:
  - “My application must be fast/secure/scale”
- Quality attributes must be **precise/measurable** for a given system design, e.g.
  - *“It must be possible to scale the deployment from an initial 100 geographically dispersed user desktops to 10,000 without an increase in effort/cost for installation and configuration.”*

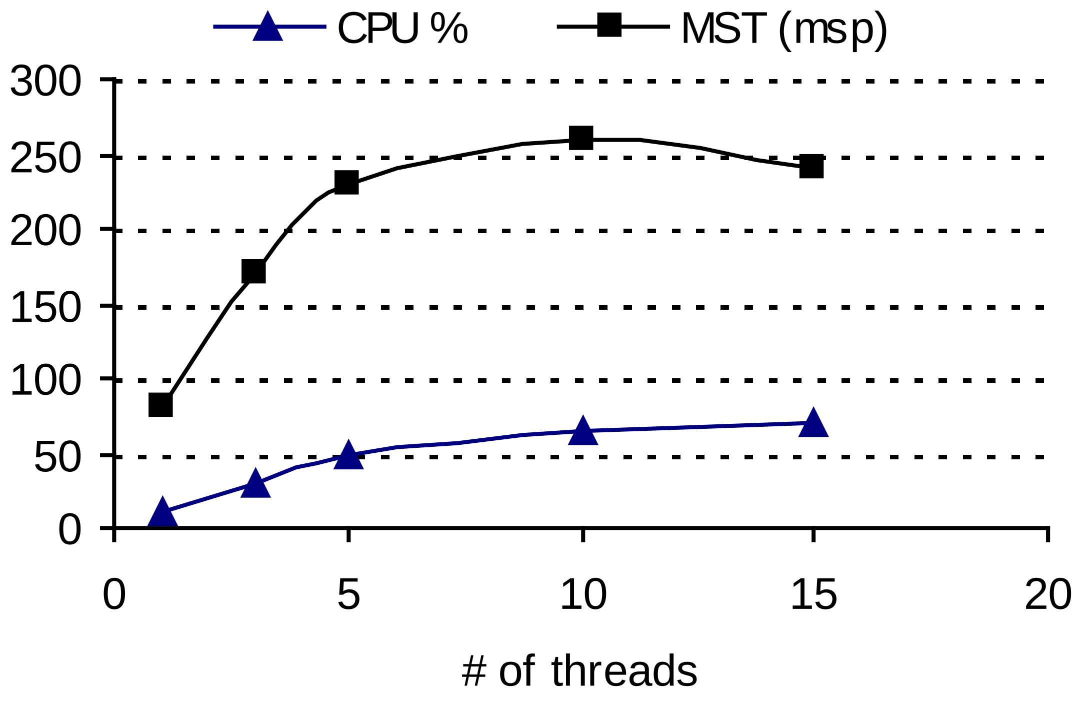
Performance

- Different ways to measure performance:
  - Throughput
  - Response Time
  - Deadlines

Performance - Throughput

- Measure of the amount of work in unit time
  - Transactions per second
  - Messages per minute

Throughput Example



Throughput Considerations

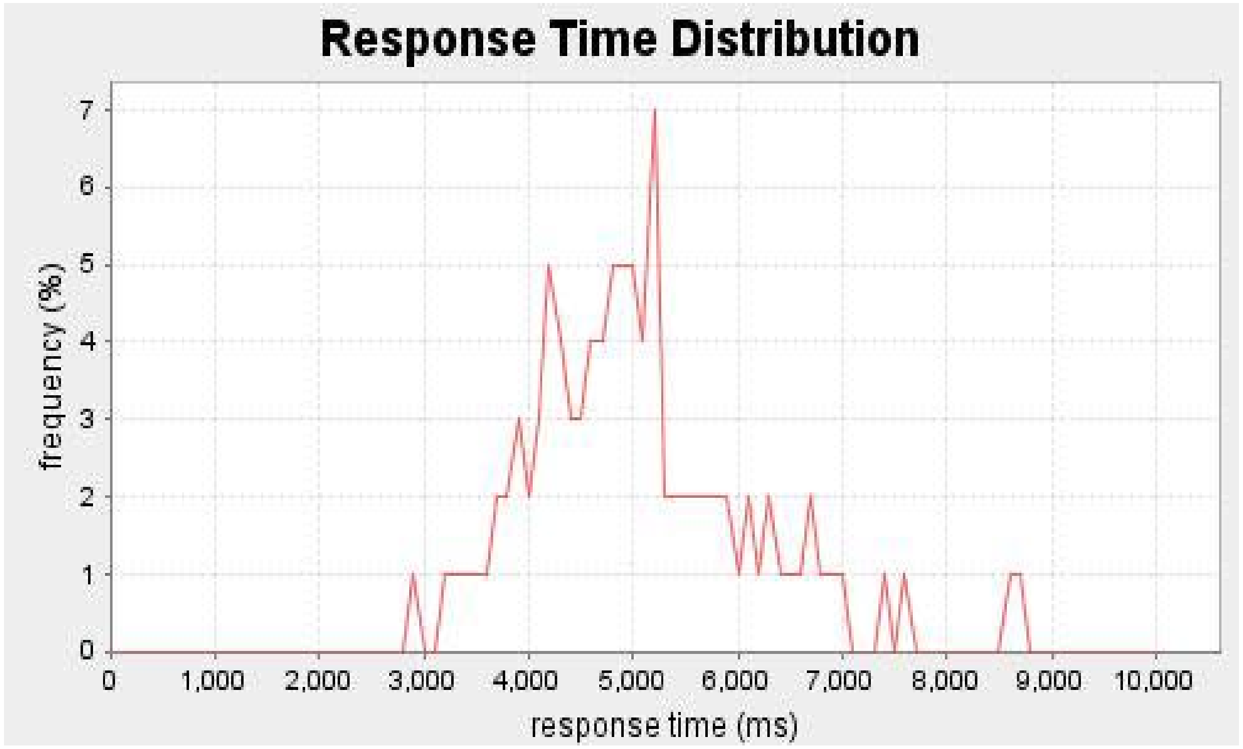
- Is required throughput:
  - Average? (Video streaming)
  - Peak? (Bidding)

- Many system have low average but high peak throughput requirements

Performance - Response Time

- **Latency or delay** an application exhibits in processing a request
  - Often an important metric for users (Point-of-sales, stock trading)

Example Response Time



- E.g. 95% of responses in sub-4 seconds, and all within 10 seconds

Response Time Considerations

- Is required response time:
  - Guaranteed? (VOIP)
  - Average? (Search)

Performance - Deadlines

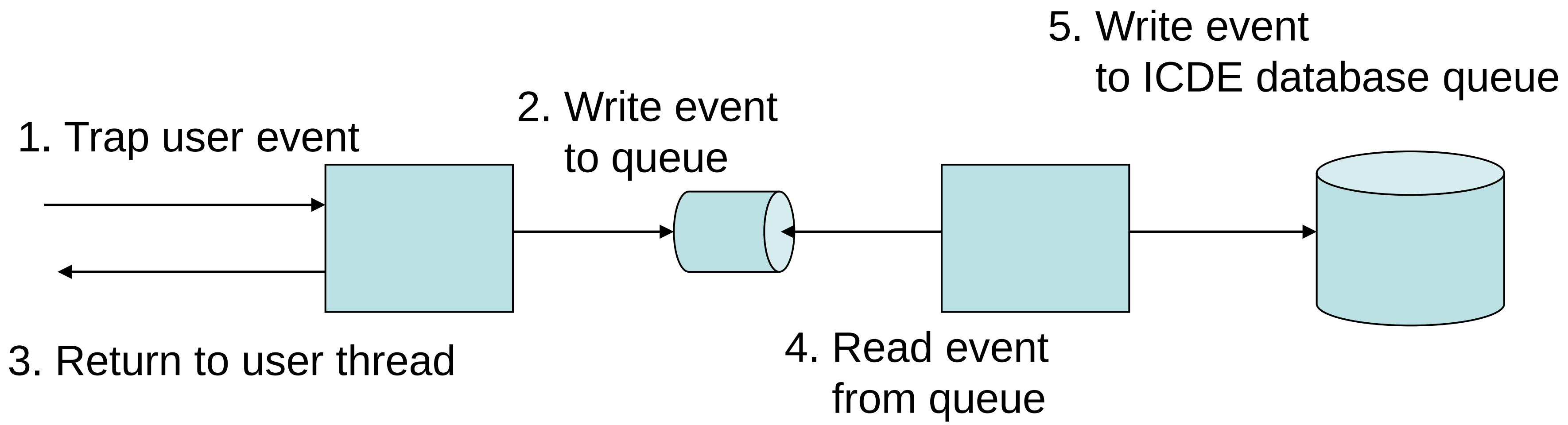
- ‘something must be completed before some specified time’
  - Payroll system must complete by 2am
  - Weekly accounting run must complete by 6am Monday

Something to watch for …

- What is a
  - Transaction?
  - Message?
  - Request?
- All are application specific measures.
- System must achieve 100 mps throughput
  - BAD!!
- System must achieve 100 mps peak throughput for *PaymentReceived* messages
  - GOOD!!!

ICDE Performance Issues

- Response time:
  - Overheads of trapping user events must be negligible to users
- Solution for ICDE client:
  - Decouple user event capture from storage using a queue



Scalability

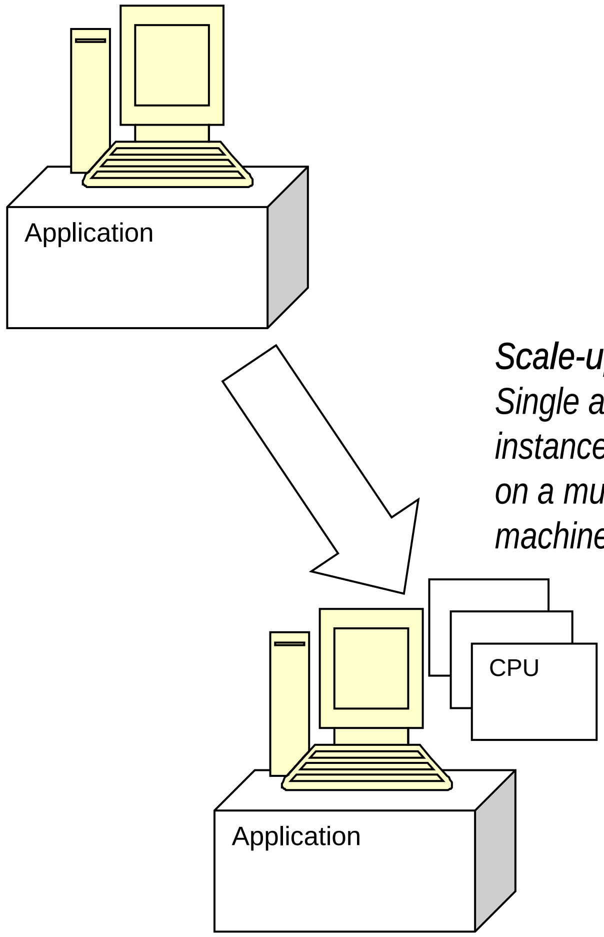
- ‘How well a solution to some problem will work when the **size of the problem increases**’
  - Request Load
  - Connections
  - Data size
  - Deployment

Scalability – Request Load

- How does an 100 TPS application behave when simultaneous request load grows?

- Ideal solution:
  - as the load increases, throughput remains constant (i.e. 100 tps), and response time per request increases only linearly (i.e. 10 seconds).

Scalability – Add more hardware



Scale-up:
Single application instance is executed on a multiprocessor machine

Scalability - reality

- Decrease in throughput and exponential increase in response time.
  - increased load causes increased contention for resources such as CPU, network and memory
  - each request consumes some additional resource (buffer space, locks, and so on) in the application, and eventually these are exhausted

Scalability - connections

- What happens if number of simultaneous connections to an application increases
  - Each connection consumes a resource?
  - Exceed maximum number of connections?

Connections Example

- ISP wants to scale to 100,000 users
  - Each user connection spawned a new process for targeted ads
  - Virtual memory on each server exceeded at 2000 users
  - Tech crash, ISP out of business

Scalability – Data Size

- How does an application behave as the data it processes increases in size?
  - Chat application sees average message size double?
  - Database table size grows from 1 million to 20 million rows?
  - Image analysis algorithm processes images of 100MB instead of 1MB?

Scalability - Deployment

- How does effort to install/deploy an application increase as installation base grows?
  - Install new users?

- Solutions typically revolve around automatic download/installation
  - E.g. downloading applications from the Internet

Scalability thoughts

- Scalability often overlooked
  - Major cause of application failure
  - Hard to predict
  - Hard to test/validate
  - Reliance on proven designs and technologies is essential

Scalability for ICDE

- Should be capable of handling a peak load of 150 concurrent requests from ICDE clients.
  - Relatively easy to simulate user load to validate this

Next Class

- Monday, Sep. 19
- Modifiability
- Security
- Availability
- Integration