Software Architectural Aspects of Contemporary Computing Platforms

Prabhakar T.V.

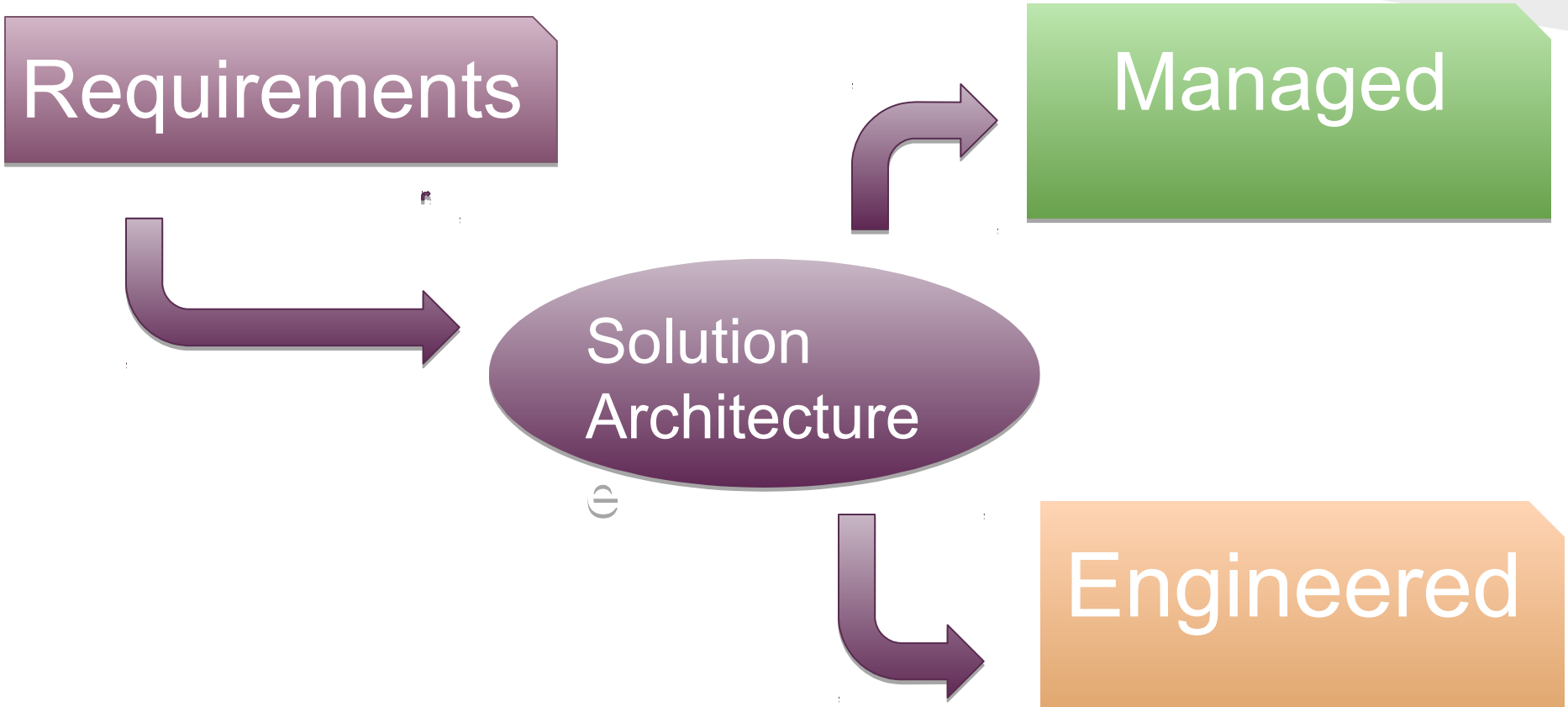
Department of Computer Science & Engineering
Indian Institute of Technology Kanpur
India
tvp@iitk.ac.in

Contemporary Computing Platform

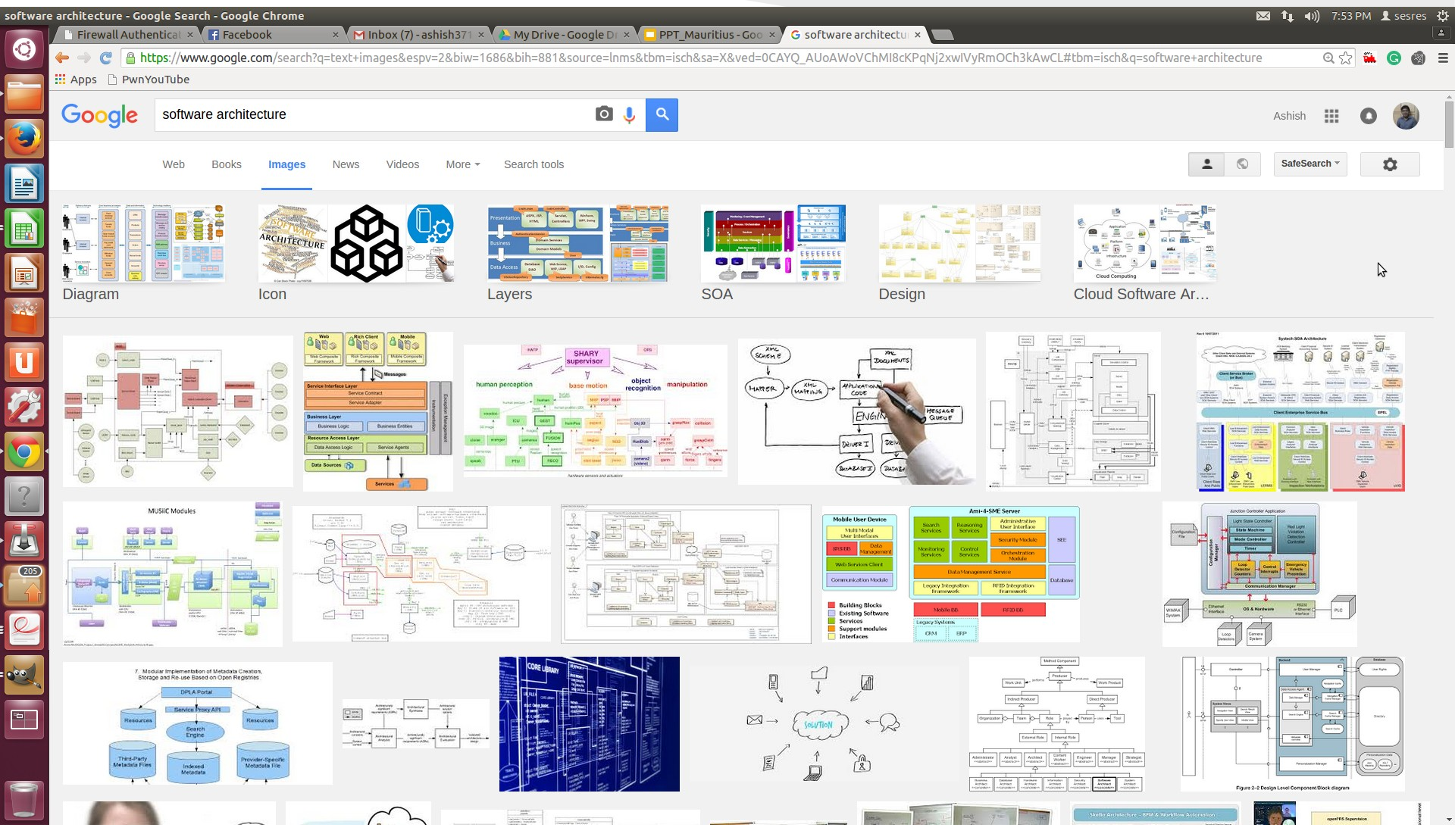
We will look at

- What is software architecture
- Mobile platforms
  - Appification process
  - Building adaptive apps using dynamic appification
- Cloud platforms
  - Introduction to cloud computing
  - Hospitality of cloud platforms
- Conclusion

Solution Architecture



What is Software Architecture?



Software Architecture

Algorithmic counterpart for large scale programs

What is Architecture?

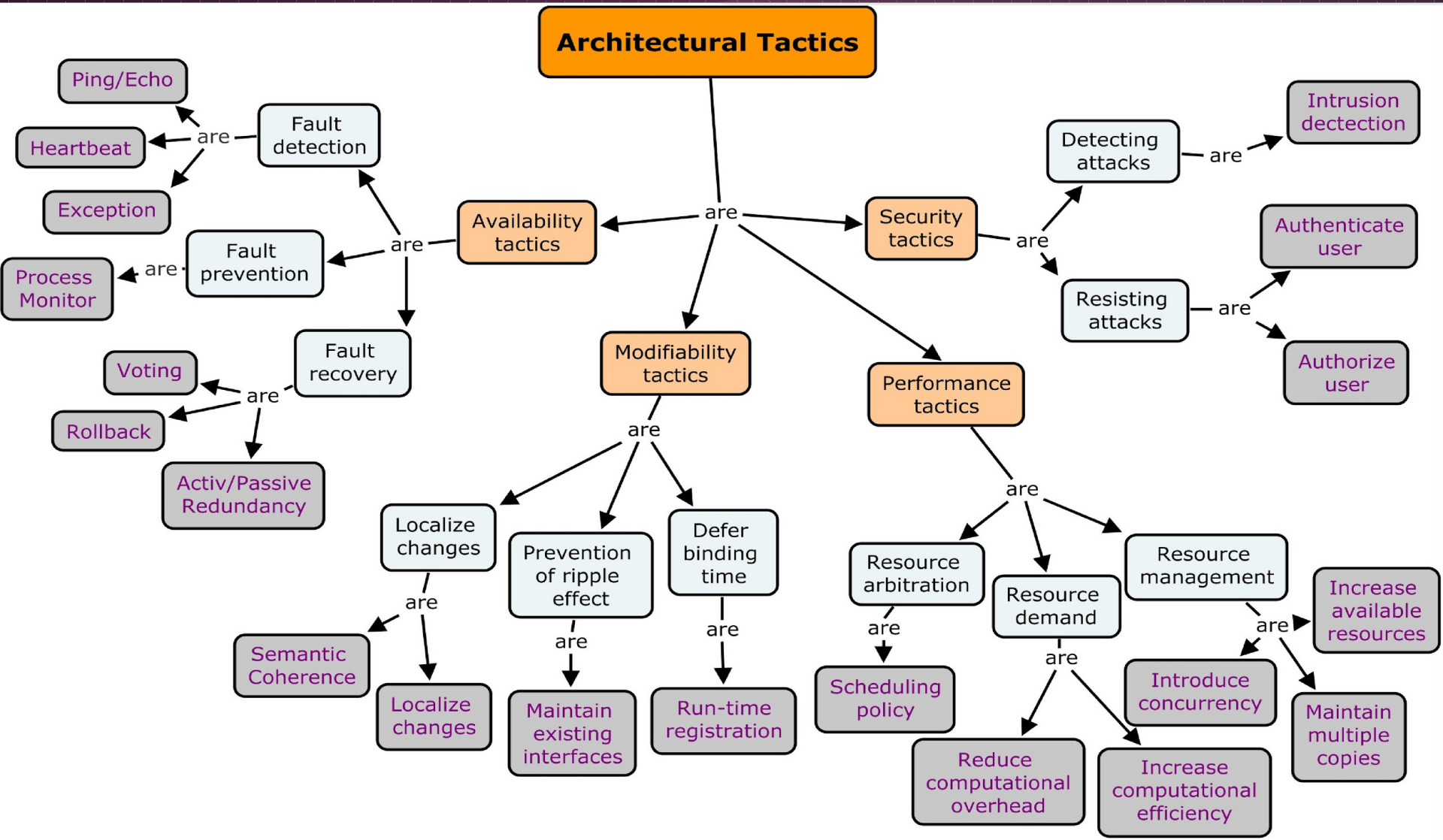
"The Software architecture of a system is the set of structures needed to reason about the system, which comprise software elements, relations among them, and properties of both"

Software Architecture in Practice
Bass, Clements, Kazman
3rd Edition

Software Architecture

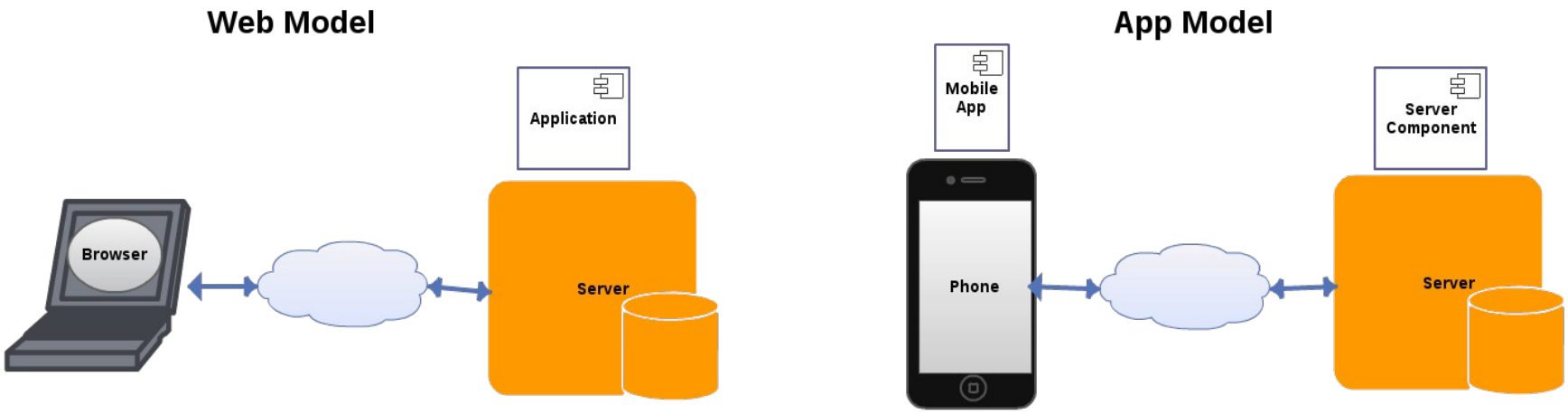
- What is it?
  - Highest level abstraction of a system
  - Represents structure and behavior of the system
- Why we need it?
  - Artifact for communication between stakeholders
  - Ensuring **Quality Attributes** e.g., availability, scalability, performance, etc.
- How do we do it?
  - Take design decisions and document them
  - Examples: what are the components/modules, who is talking to whom, where are they deployed?
- Tools: Architectural tactics and patterns

Example: Architectural Tactics



Mobile Platforms

Web Model vs App Model



Appification

- Appification
  - The process of building app-based web applications
- It can help in achieving various quality attributes like usability, performance, etc.
- It also provides an opportunity to exploit resources available at client devices
  - Operational load is partially migrated to the client devices.

Appification

- Challenges
  - Diversity and resource constraints
  - Dynamic environments: Availability of resources at client devices can vary with time like battery, network, etc.
  - Issues in ensuring quality attributes of the application – client perspective
  - Limits the ability to exploit client resources – server perspective

Research Problem

- How to design and build applications to accommodate the dynamic environments while considering both client and server perspectives?
  - Application should be able to manage its expectations from the environment
- Directions
  - Focus is on investigating the architecture-level decisions that are specific to the appification process

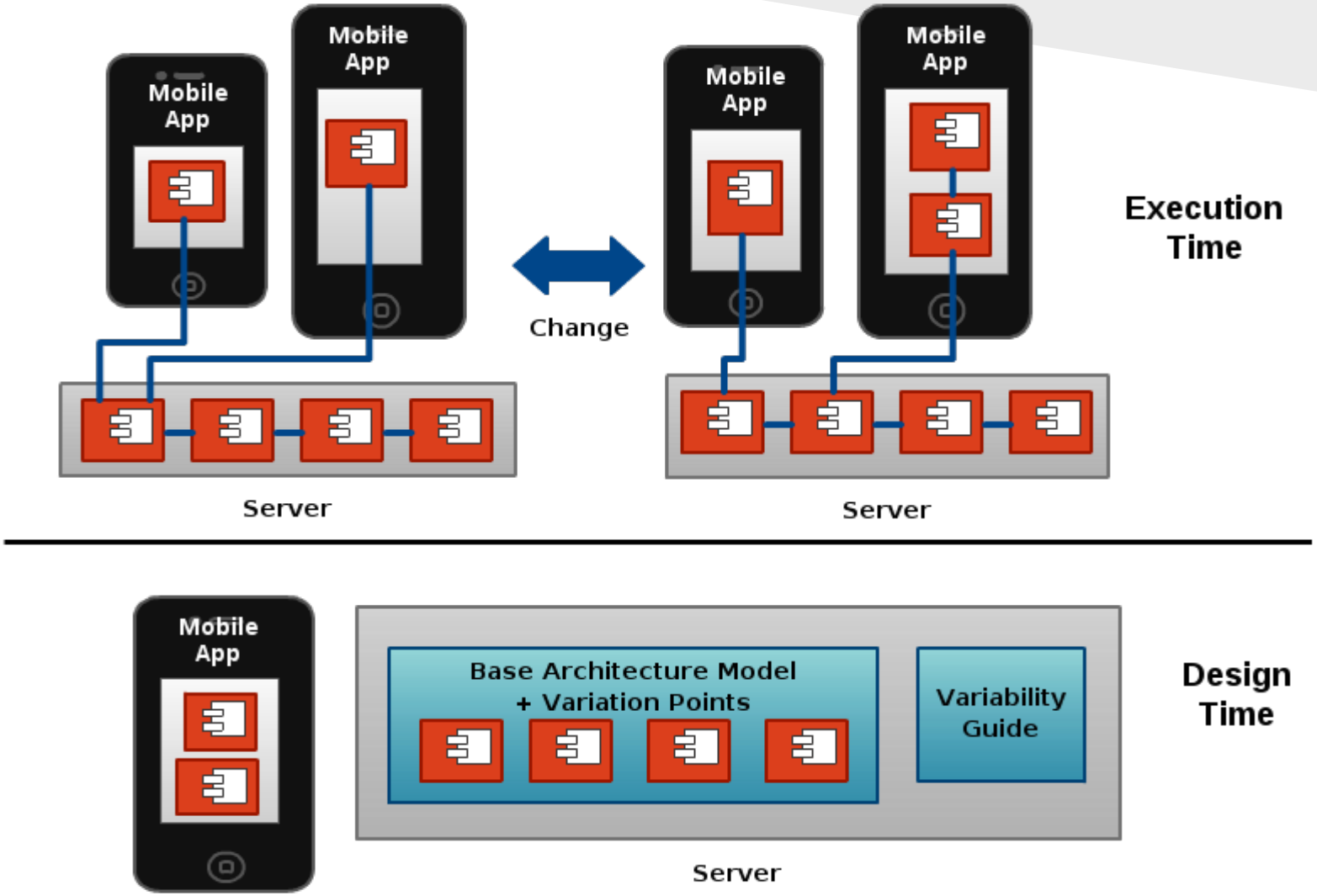
Key Observations

- Appification Strategy
  - Divide the application components into two groups; one to be executed on the client device and the other on the server
- Selection of an Appification Strategy:
  - Trade-off between multiple quality attributes
  - In current approaches, performed during the design phase

What we can do?

- Dynamic Appification
  - Appification Strategy of the application is not fixed at design-time and can vary at run-time
- Approach
  - Application is designed to incorporate multiple appification strategies by introducing variability in the architecture
  - At run-time, the application can select the suitable variant for client(s) depending upon the environment

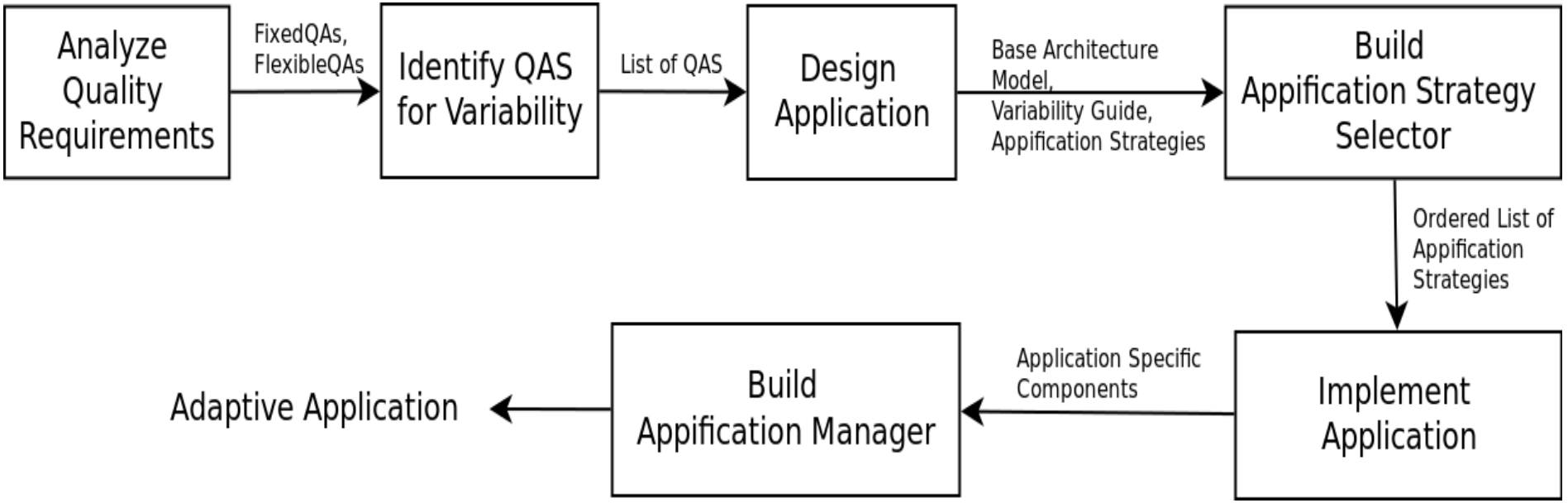
Adaptation Example



Dynamic Appification

- How to build adaptive application using dynamic appification?
  - Need the ability to change appification strategy
  - Need the ability to adapt in dynamic environments
  - ……
  - What are the responsibilities and how to handle them?

Appification Framework



Appification Framework

- **Step1**: Analyze Quality Requirements
  - What are the quality attributes that allow/restrict variations in their expectations ?
  - Ex: Response-time can vary between 0-10 seconds - there can be multiple strategies satisfying this variation
- **Step2**: Identify QAS for Variability
  - When should it adapt and what should be achieved?
  - Requirements are represented as Quality Attribute Scenarios
  - Ex: If battery power is less than 30% of full power, reduce energy consumption on the client

Appification Framework

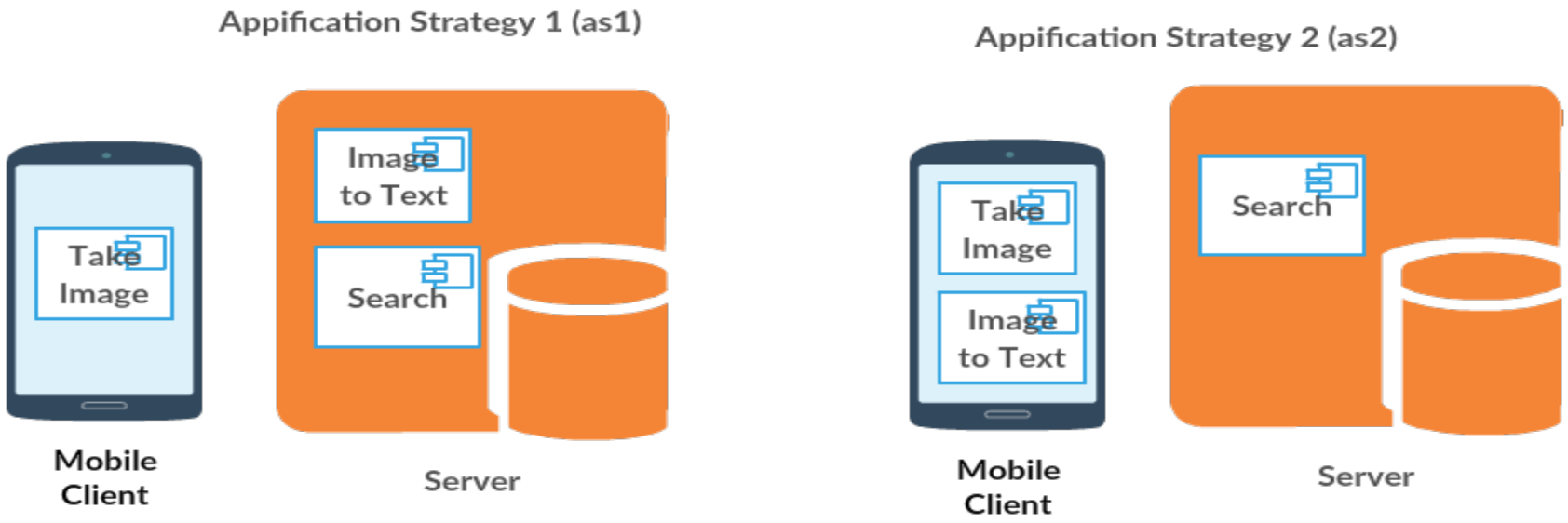
- **Step3**: Design Application
  - Identify useful appification strategies by analyzing components.
  - Evaluate strategies in terms of their impact on QAs.
  - Design base architectural model with variation points
    - A variation point for a component has two choices: whether to execute that component on client device or on server.
- **Step4**: Build Appification Strategy Selector
  - Which appification strategy should it adapt to? – MCDM Problem (TOPSIS)
  - Minimize the adverse effect on other quality attributes

Appification Framework

- **Step5**: Implement the Application
  - How to minimize overhead of changing appification strategies?
  - Code redundancy - deploy code on both client and server
  - Defer binding - bindings between components can be changed at run-time
- **Step6**: Build Appification Manager
  - How will the application adapt?
  - Need a system to manage the application
  - MAPE loop - Context Monitor, QAS Anaylze, Strategy Planner and Executor

Case Study

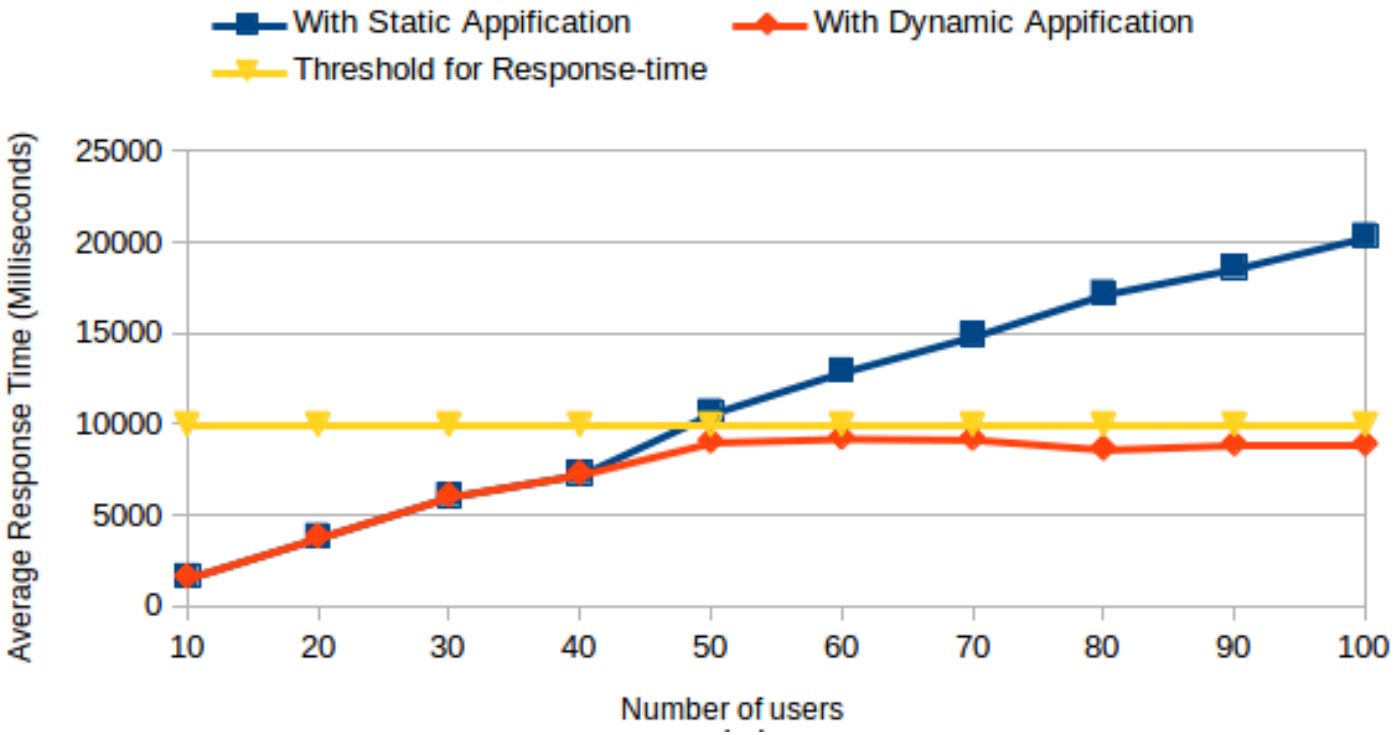
- Simple application that facilitates image-based searching/translation… of products



Adaptations

- Server driven adaptation
  - If the server reaches 85% of its capacity, reduce the load on the server.
    - Capacity: Number of clients the application can serve while maintaining the average response time to less than 10 seconds
- Client-driven adaptation
  - If the battery power at a client is less than 30% of full power, reduce energy consumption at the client device
  - If the client is having intermittent network connectivity, reduce response-time from the server so that the dependency on a stable network is narrowed

Server-driven adaptation results



Cloud Platforms

Contemporary Platforms

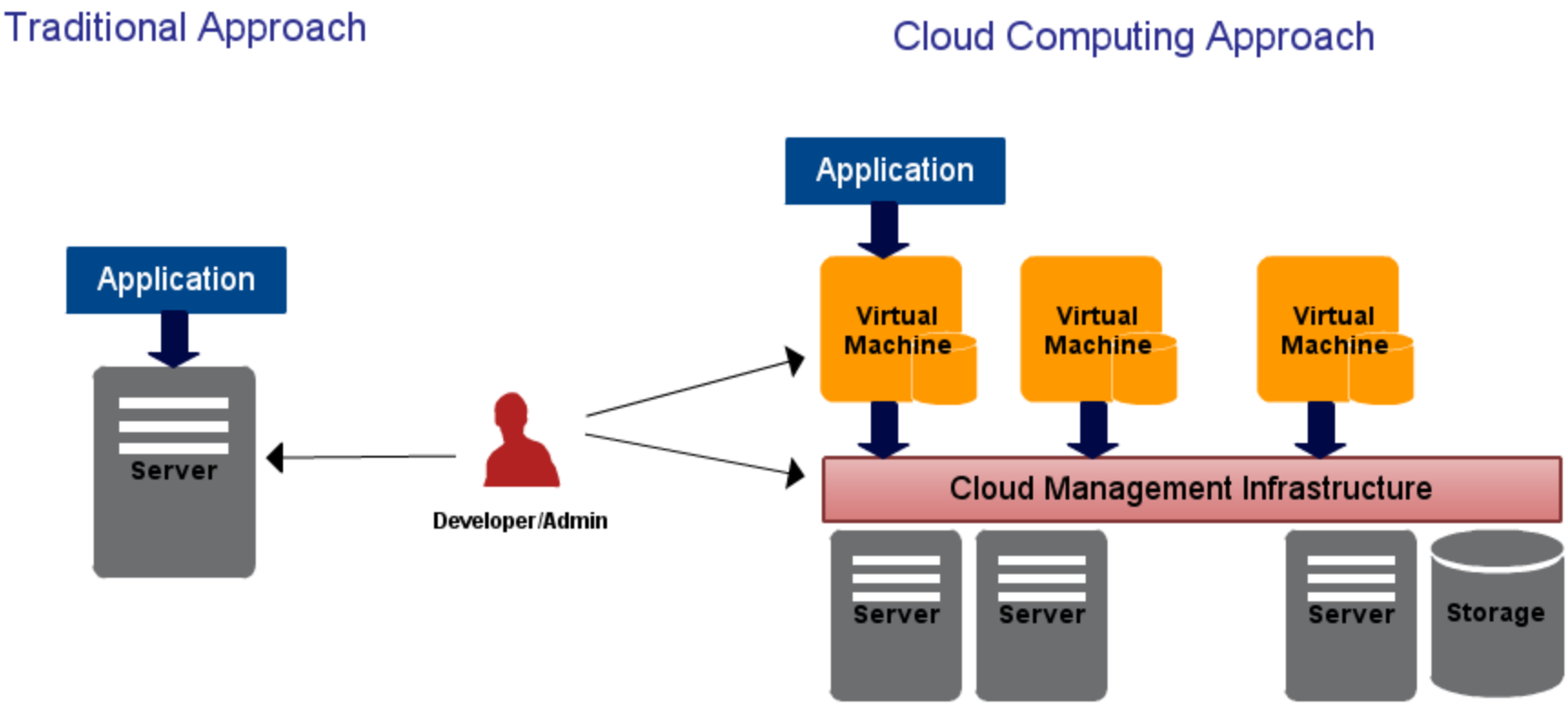
- While designing applications for these platforms, it is desired to identify:
  - Impact of the platform on application's quality attributes.
  - New ways to exploit the platforms to ensure desired quality attributes.

Cloud Platforms

- Rent a machine - IaaS
- Rent a platform - PaaS
- Rent a service – SaaS

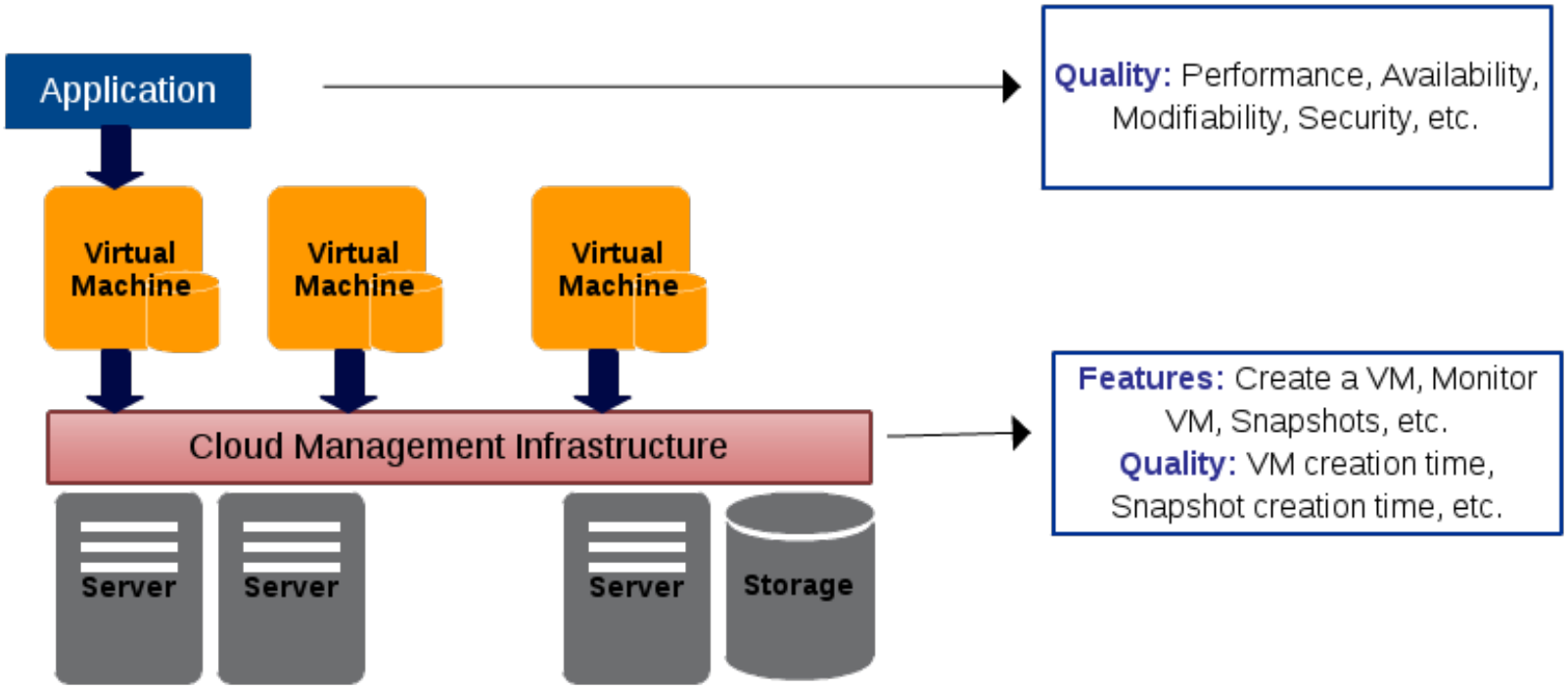
- Public Cloud, Private Cloud

Cloud Computing



Impact on the Application?

- Deploying on a cloud does not ensure application level quality attributes



Quality

Quality Not automatic

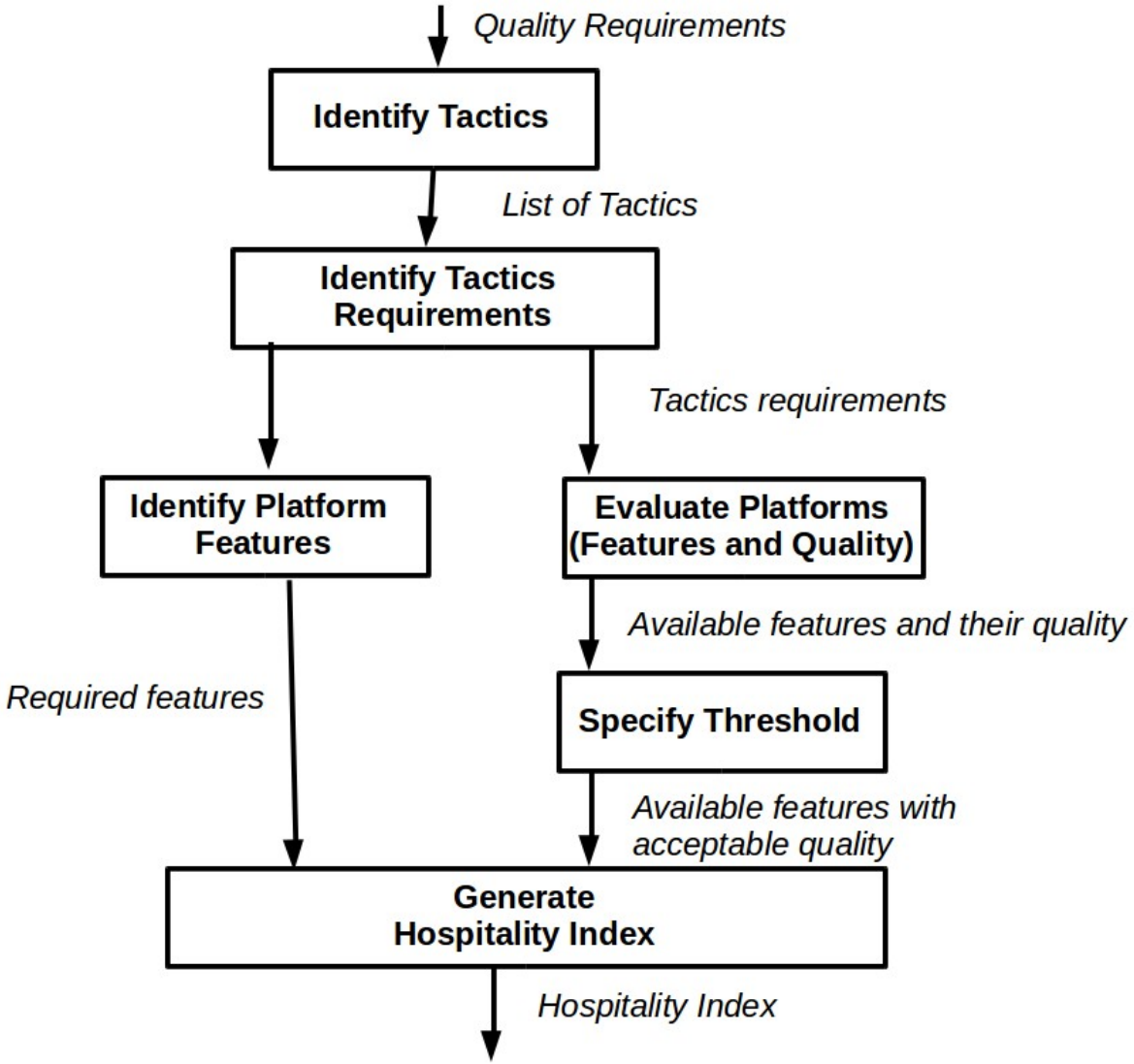
Hospitality

- **Hospitality** of a cloud platform represents support provided by it to the applications for achieving quality requirements.
  - Higher hospitality for a platform towards a quality attribute makes it possible and easy for the application developer to achieve that quality in the application.
  - Lower hospitality means that the developer has to create extra resources/components at the application level.

Hospitality Framework

- Methodological framework for investigating hospitality of cloud platforms towards quality attributes of applications.
  - Scope is limited to IaaS platforms.
  - Output is an index (Hospitality Index) for representing hospitality.
  - Investigated hospitality of two open-source IaaS platforms; Eucalyptus and OpenNebula.

Methodology of Hospitality Framework



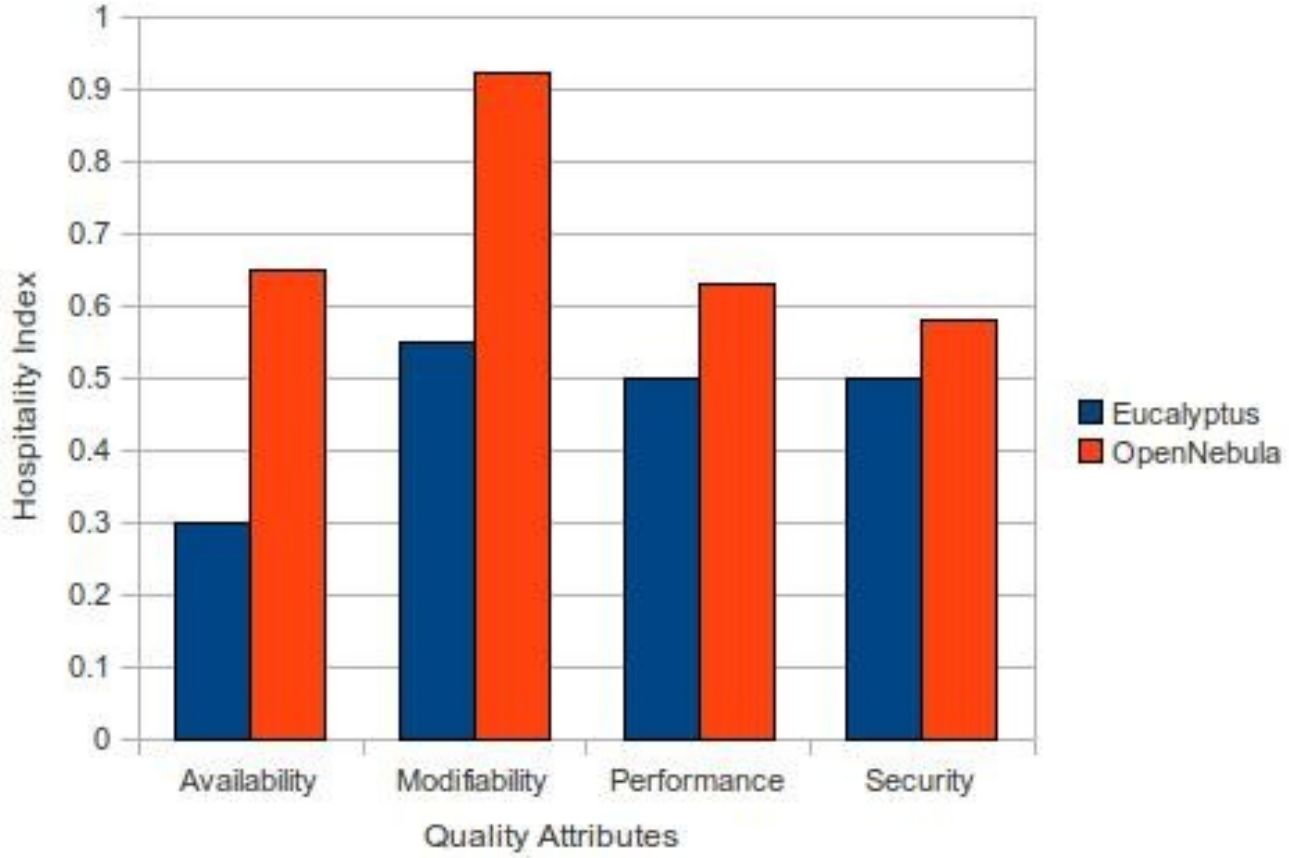
Example

- Quality Attribute: Availability
  - Identify Tactics: Ping&Echo
  - Identify Tactics Requirements:
    - Isolated components for monitoring components (PingSender, Monitor)
  - Identify Platform Features:
    - Ability to create multiple VMs in separate cluster, network, hardware
  - Specify Threshold
    - Creating a VM should not take more than 45 seconds

Hospitality Index- Tactic Level

QA	Tactics	Platform Features (available for applications)	Features Availability		Hospitality Index	
			Euca-lyptus	Open Nebula	Euca-lyptus	Open Nebula
Availability	Ping & Echo, System Monitor	Ability to create multiple VMs in separate clusters	✓	✓	0.25	0.75
		Ability to create multiple VMs in separate networks	✘	✘		
		Ability to create multiple VMs on separate hardware	✘	✓		
		API for monitoring VM's state	✘	✓		
	Voting	Ability to create multiple VMs with different hardware specifications (CPU, Memory, Storage)	✓	✓	0.4	0.8
		Ability to create multiple VMs in separate clusters	✓	✓		
		Ability to create multiple VMs in separate networks	✘	✘		
		Ability to create multiple VMs on separate hardware	✘	✓		
		API for monitoring VM's state	✘	✓		
	Active/Passive Redundancy, Spare	Dynamic binding of IP Address (Elastic IP)	✓	✓	0.37	0.87
		Dynamic binding of storage devices	✓	✓		
		Run-time sharing of storage devices between VMs	✘	✓		
		Ability to control VM scheduling	✘	✓		
		Ability to create multiple VMs in separate clusters	✓	✓		
		Ability to create multiple VMs in separate networks	✘	✘		
		Ability to create multiple VMs on separate hardware	✘	✓		
		API for monitoring VM's state	✘	✓		
	Rollback	Ability to create snapshot and rollback a VM	✓	✓	0.5	0.5
		Snapshot of VM with active memory	✘	✘		
	Process Monitor	API for monitoring VM's state	✘	✓	0	0.33
		Ability to register for platform failure events	✘	✘		
		API for monitoring network, storage	✘	✘		

Hospitality Index -QA Level



Framework Utilities

- Platform Selection
  - Select the cloud platform with highest hospitality.
  - Multi-attribute decision problem as multiple QAs.
  - We used *Technique for Order of Preference by Similarity to Ideal Solution (TOPSIS)* framework.
- Selection of architectural components
  - While developing application for a particular cloud, use the components that have high hospitality on that platform.
- Improve hospitality of a cloud platform
  - Identify features of cloud platforms which have maximum impact on hospitality towards various tactics

Conclusions

- Underlying Cloud and Mobile platforms directly affect application design process and quality attributes.
  - We need new tools and techniques to investigate such impact and build quality applications.
- Support from cloud can be investigated in terms of Hospitality.
- Dynamic Appification can help in achieving various quality requirements in dynamic environments of mobile devices.

Research Directions

- New Quality Attributes?
  - Identify new patterns/tactics
- Green Computing
  - Energy-Efficiency as a QA
- What about PaaS, SaaS?
- MicroServices

References

- Ashish Agrawal and T. V. Prabhakar. 2013. Hospitality of cloud platforms. In *Proceedings of the 28th Annual ACM Symposium on Applied Computing* (SAC '13), Portugal.

- Ashish Agrawal and T.V. Prabhakar. 2015. Towards a Framework for Building Adaptive App-based Web Applications using Dynamic Appification. In Proceedings of the 9th European Conference on Software Architecture (ECSA’15), Croatia.

Thank You

- Questions??

Prabhakar T.V. (tvp@iitk.ac.in)

Questions

- Questions faced by an architect:
  - Which cloud platform should be used to deploy the application with desired quality attributes?
  - What architectural components should be used to design an application for a particular cloud?

- [Research Question]: How to investigate the impact of underlying cloud platform on the application design process and on the quality attributes of the application?

Appification Manager