

# Context Awareness in Mobile Systems (2)

CSE 162 – Mobile Computing

Hua Huang

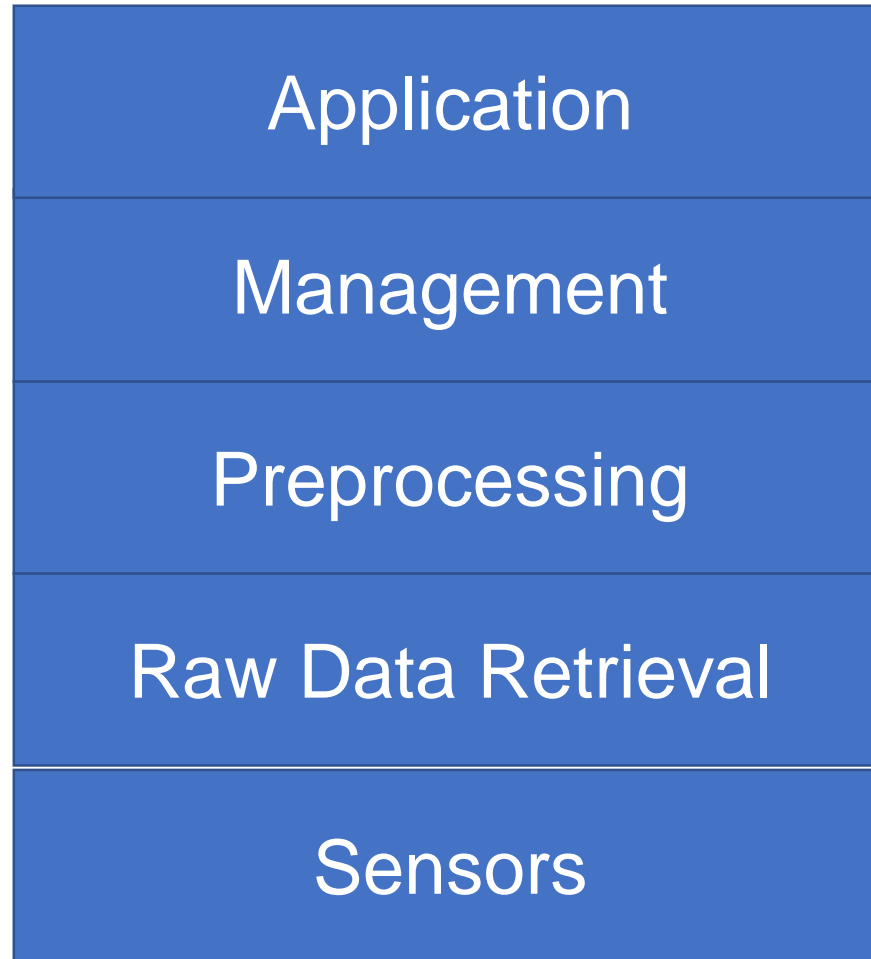
Department of Computer Science and Engineering

University of California, Merced

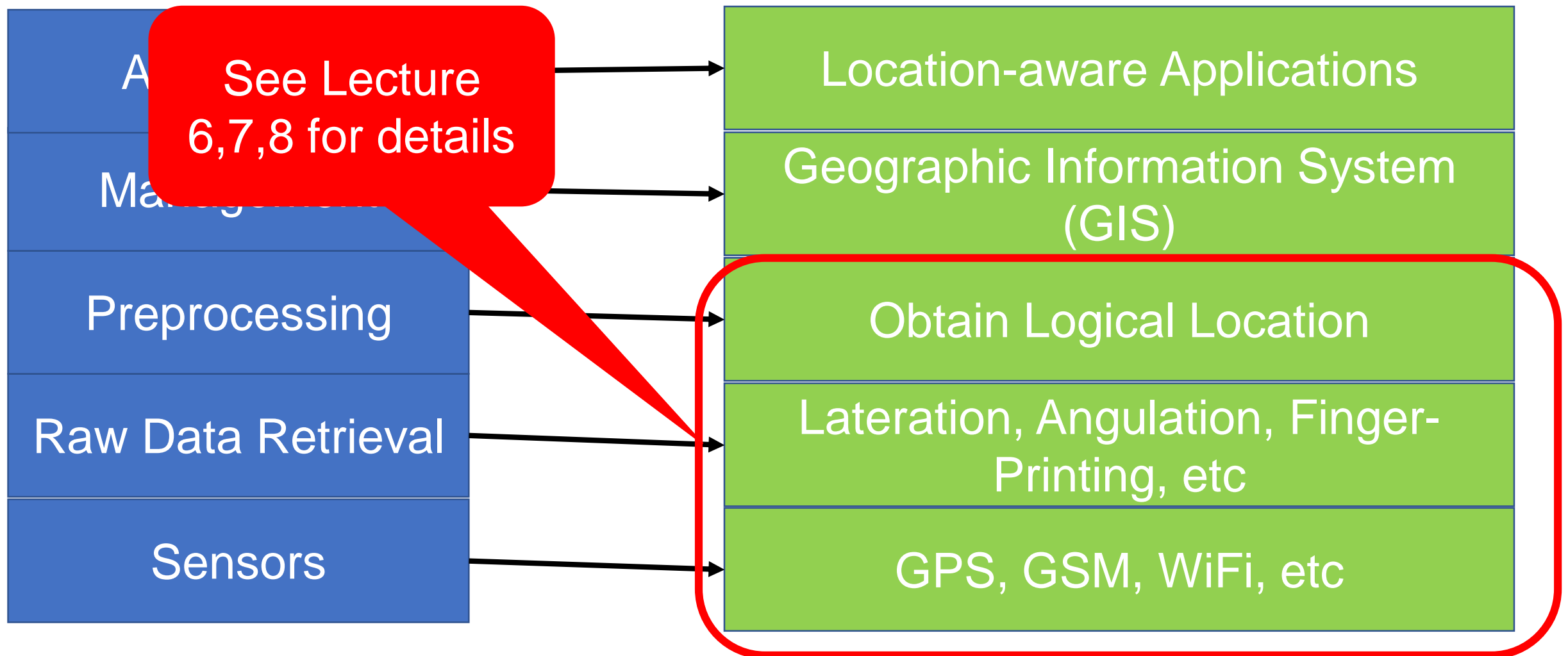
# Agenda: context aware system frameworks

- Framework for location (spatial) awareness
- Framework for device UI context awareness
  - Techniques for UI adaptation in Android
- Framework for network awareness
  - Adaptive video streaming

# Recap: the general framework for context aware systems



# Overview for location aware systems

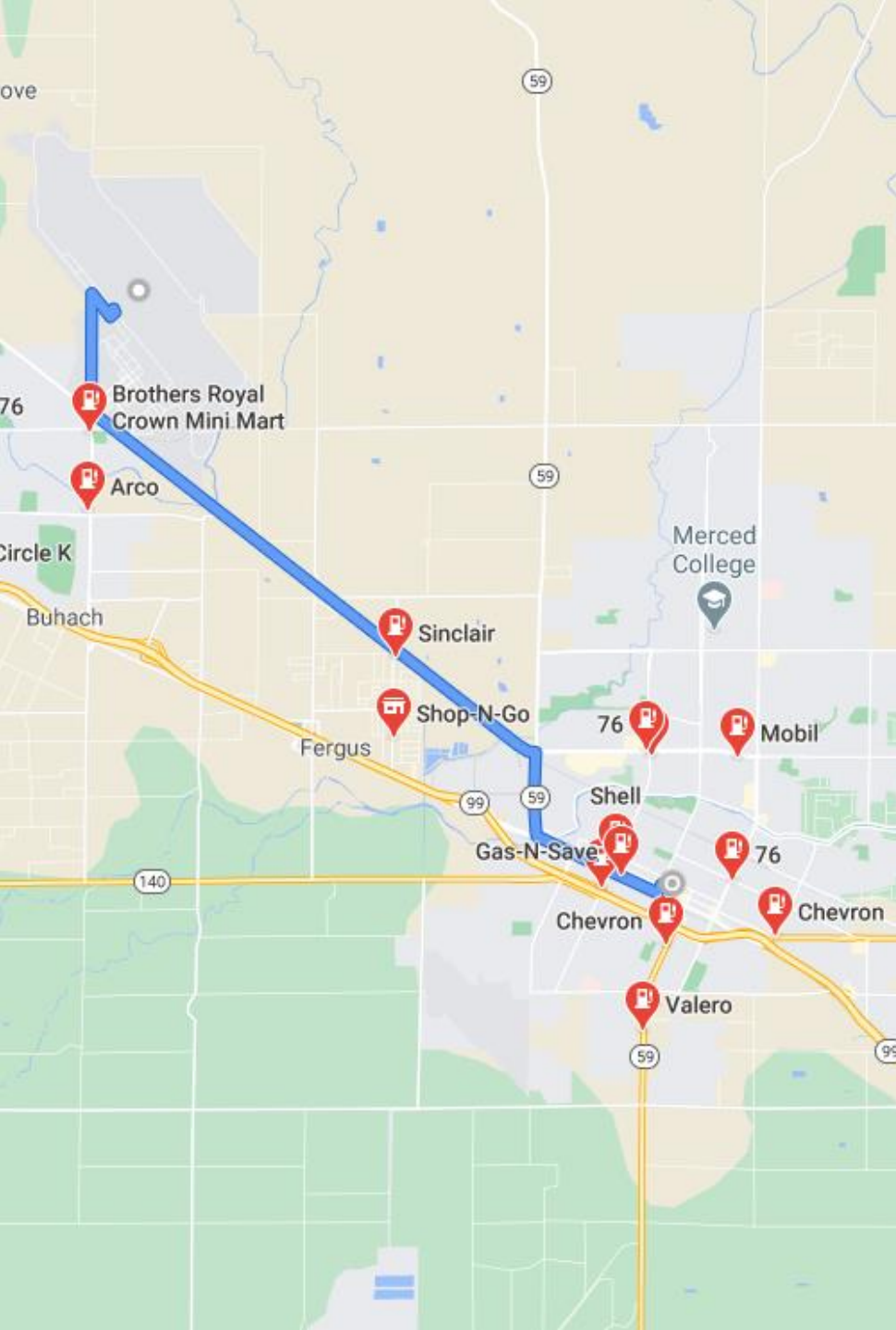


# Application: adapting to spatial viewpoints

- Show a selection of restaurants within a region or show a route between a location and a destination
- Dynamically create maps to show only the layers and objects of interest
  - E.g., show walking path only
- Adapt the map for difference displays and network connection
  - Different screen sizes
  - Pre-cache maps or on-demand maps

# Geographic Information System (GIS)

- Challenges in managing location contexts
  - How to search and match location contexts?
  - How to store and share location contexts?
- The determination of a higher-level query relationship, such as what is the next petrol stations around this route, requires different kinds of algorithms
  - Search space can be very large
- Solution: spatial data structures: enable search and organization of location info

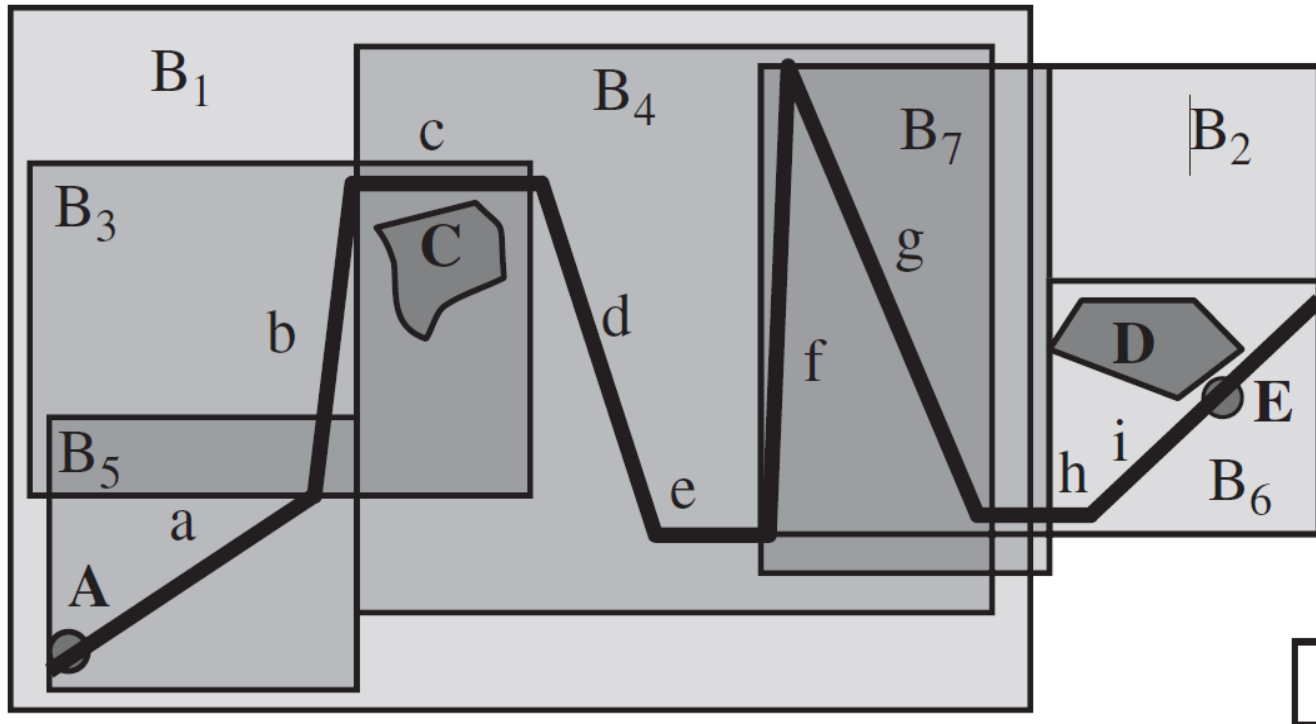


# Example: gas stations

---

- # of gas stations in the area: 84
- # of gas stations near the route: 5
- Challenge: given the route, how to quickly find the relevant points of interest?

# Location context query and management: algorithm and data structure

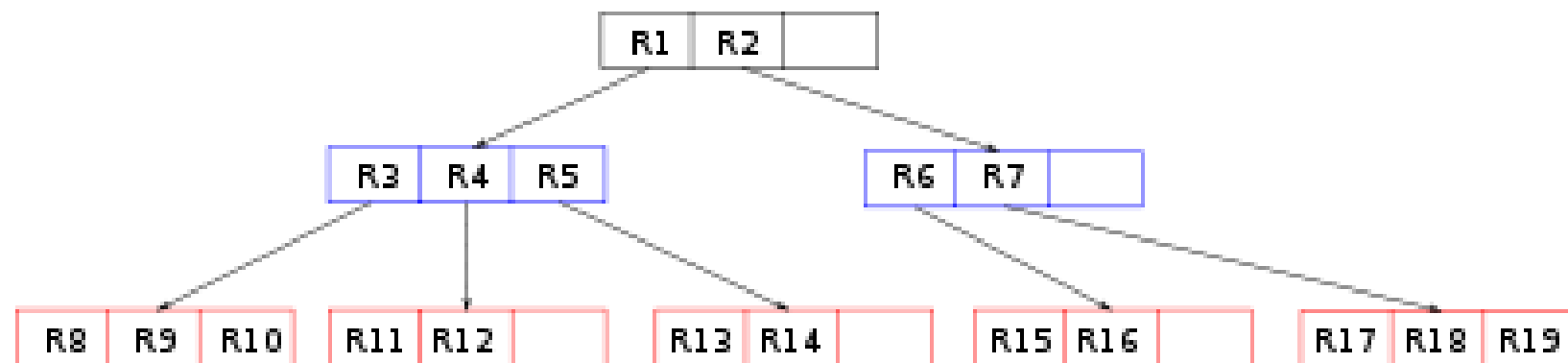
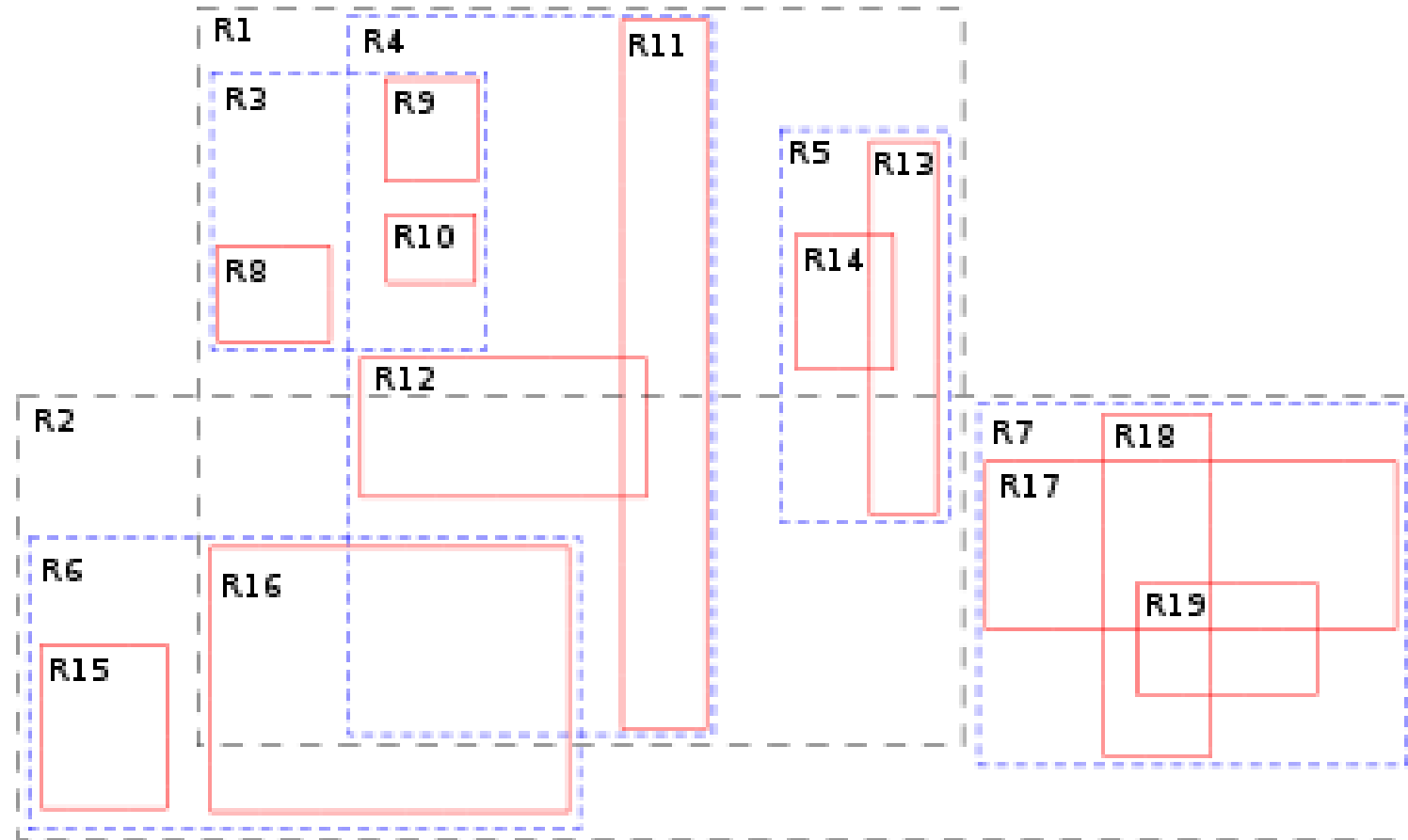


- Elements in a map
  - Road segments: a-i,
  - Buildings: C and D
  - Bounding boxes B1-B7
- Spatial query:
  - Find the route from C to D
  - Find bounded boxes that contain the route
  - Find the objects near the route



# Location context query and management: algorithm and data structure

- Solution: R-Tree
  - Group nearby objects and represent them with their minimum bounding rectangle in the next higher level of the tree
  - At the leaf level, each rectangle describes a single object;
  - At higher levels, the aggregation includes an increasing number of objects. Can also be seen as an increasingly coarse approximation of the data set.



# User Interface Context Awareness

# Motivation

- Vast diversity of output and input capabilities in mobile devices



# UI Profile Acquisition

- The World Wide Web Consortium (W3C) has defined CC/PP: the Composite Capabilities / Preferences Profile
  - defines a client profile data format, and a framework for incorporating application and operating environment-specific features including the terminal hardware, the terminal software and the terminal Web browser.

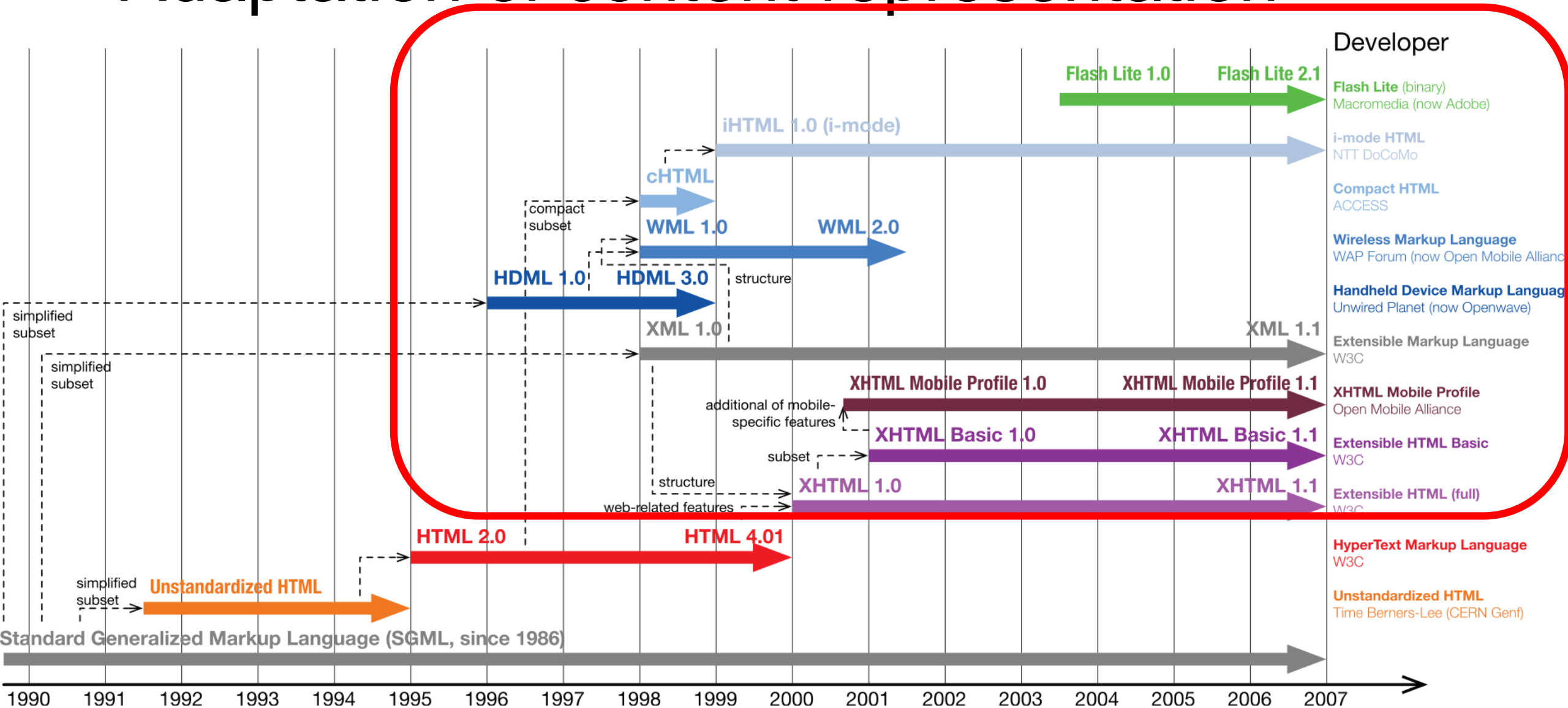
# UI Content Adaptation

- Adaptation of content representation
- Adaptation of interaction
- Adaptation of content
- Adaptation of presentation style

# Adaptation of content representation

- Many variations of HTML were designed for mobile devices
  - Stripped down versions of HTML that consume less resources

# Adaptation of content representation





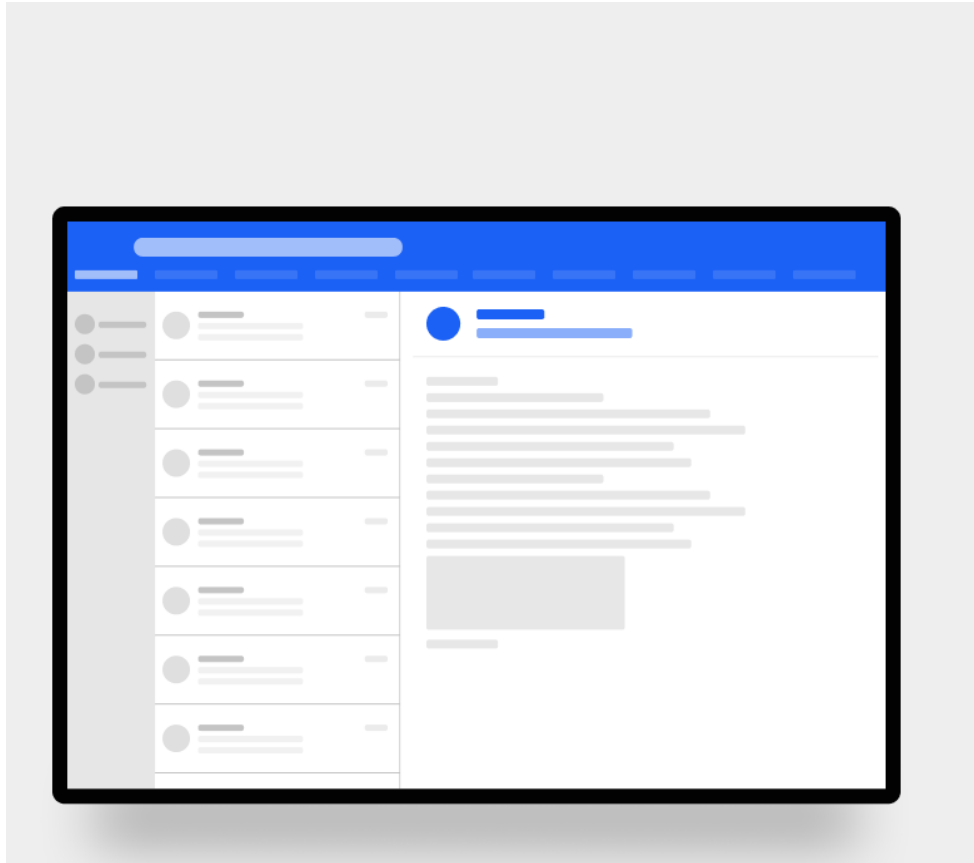
# Adaptation of interaction

- World Wide Web Consortium has created a Multimodal Interaction Activity
  - Allow users to select the most appropriate interaction, whilst enabling developers to provide an effective user interface
- Users will be able to
  - input via speech, handwriting, and keystrokes
  - output presented via displays, pre-recorded and synthetic speech, audio, and tactile mechanisms such as mobile phone vibration

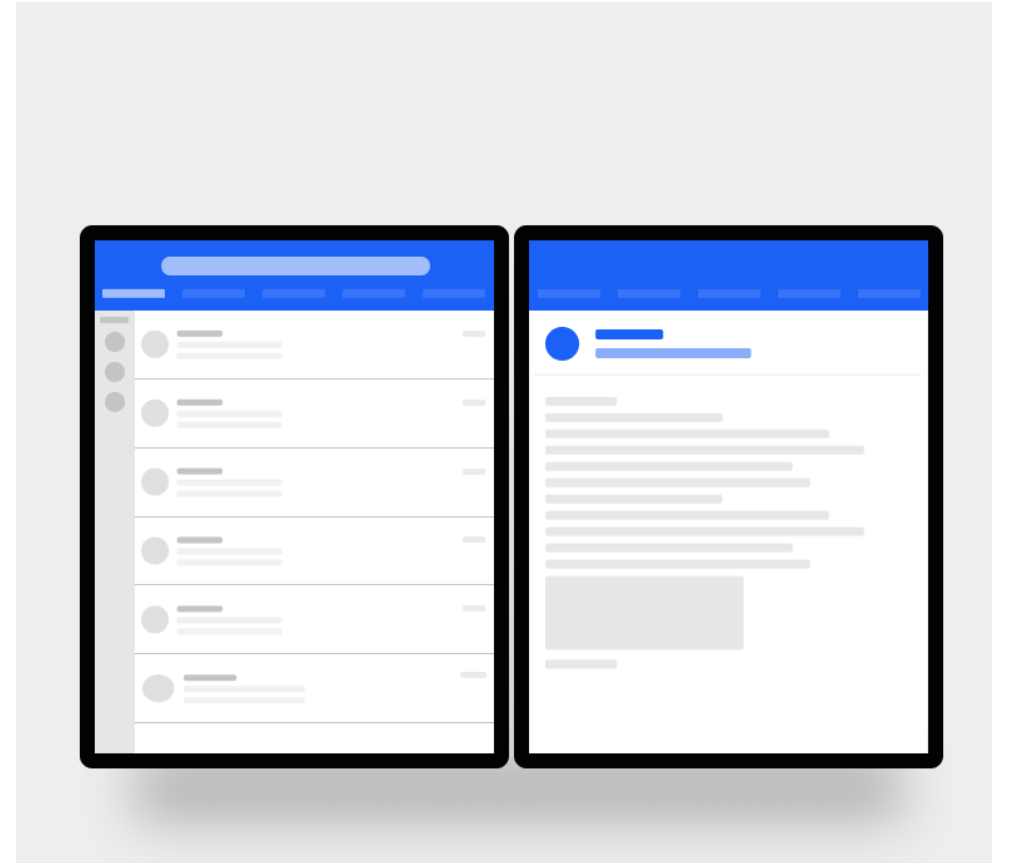
# Adaption of contents

- Example: to display a large map on a small screen.
- Often involves more than simple scaling because vital detail may be lost when content is reduced.
- Common approaches
  - split the big content into multiple screens and to support techniques to navigate between them (stacking or scrolling windows).
  - Reduce full text to only the title or a summary

# Example: stacking windows

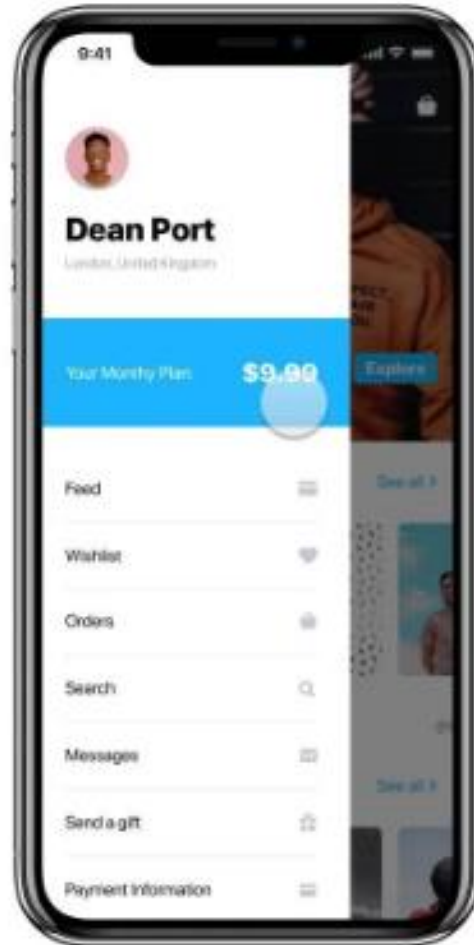


Large screen



Small screen

# Example: scrolling window



# Adaptation of presentation style

- Adapt to the layout for a given device profile
  - e.g. portrait vs landscape



# Strategies to adapt to different device capabilities

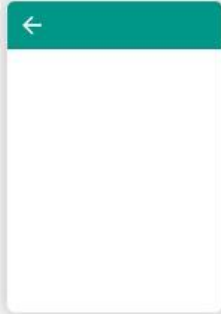
- Lowest Common Denominator (LCD) approach:
  - Create contents for a few categories of devices
  - Each device in the category supports a lowest common denominator profile.
- Transcoding of content to adapt it to specific types of access devices
  - this transforms content from one form to another via clearly defined mapping functions.

# LCD Approach

ANDROID PLATFORM VERSION	API LEVEL	CUMULATIVE DISTRIBUTION
2.3 Gingerbread	10	
4.0 Ice Cream Sandwich	15	97,4%
4.1 Jelly Bean	16	95,2%
4.2 Jelly Bean	17	87,4%
4.3 Jelly Bean	18	76,9%
4.4 KitKat	19	73,9%
5.0 Lollipop	21	40,5%
5.1 Lollipop	22	24,1%
6.0 Marshmallow	23	4,7%

Create New Project

## Configure Your Project



Empty Activity

Creates a new empty activity

Name

My Application

Package name

com.example.myapplication

Save location

C:\Users\mmrah\AndroidStudioProjects\MyApplication2

Language

Kotlin

Minimum SDK

API 14: Android 4.0 (IceCreamSandwich)

**i Your app will run on approximately 100% of devices.**  
[Help me choose](#)

☐ Use legacy android.support libraries ?

Previous

Next

Cancel

Finish

# UI Compatibility in Android

- Android device diversity:
  - Screen sizes
  - Pixel densities
  - Wear OS, TV, Auto, and Chrome OS
  - Languages

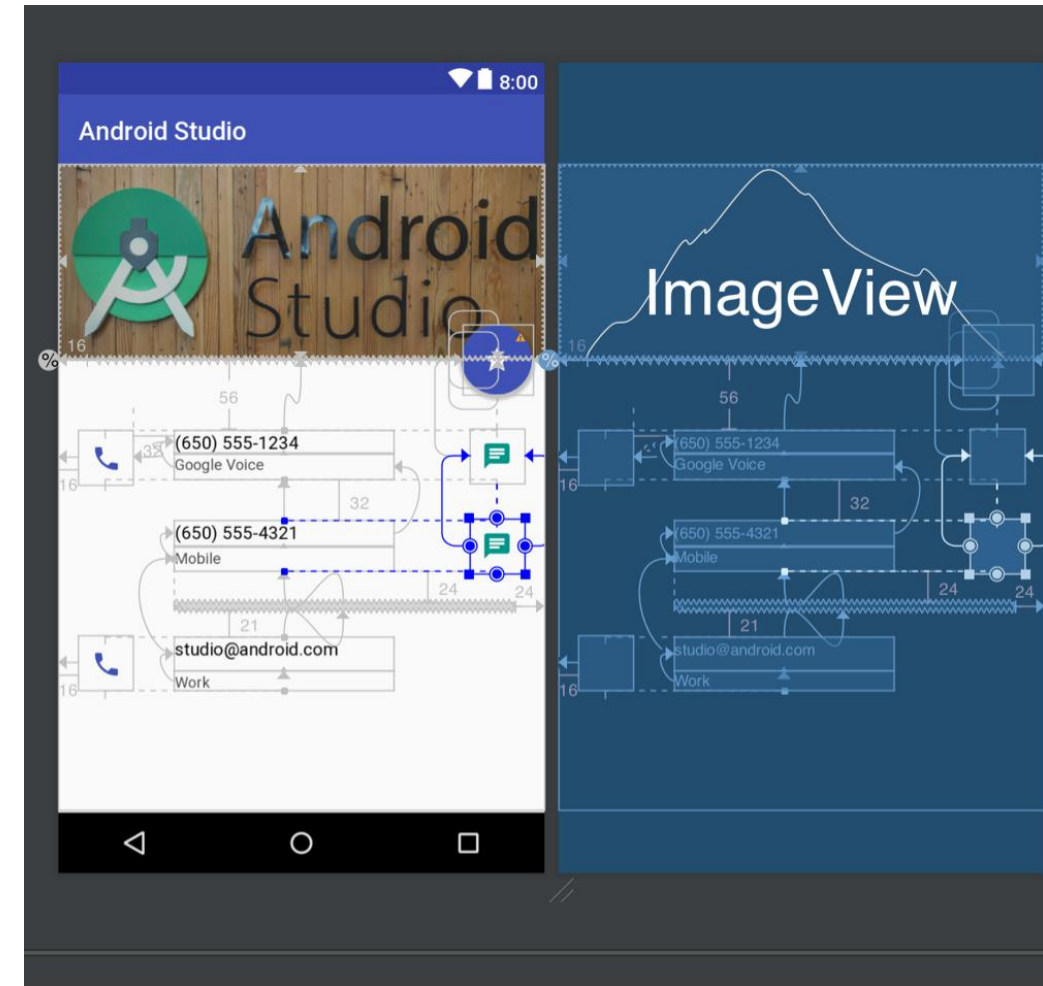


# Techniques to support different screen sizes

- Create a flexible layout
  - Use `ConstraintLayout`
  - Avoid hard-coded layout sizes
- Create alternative layouts
- Create stretchable nine-patch bitmaps

# Create Flexible Layout

- Use ConstraintLayout
  - ConstraintLayout allows you to specify the position and size for each view according to spatial relationships with other views in the layout. This way, all the views can move and stretch together as the screen size changes.



# Create Flexible Layout

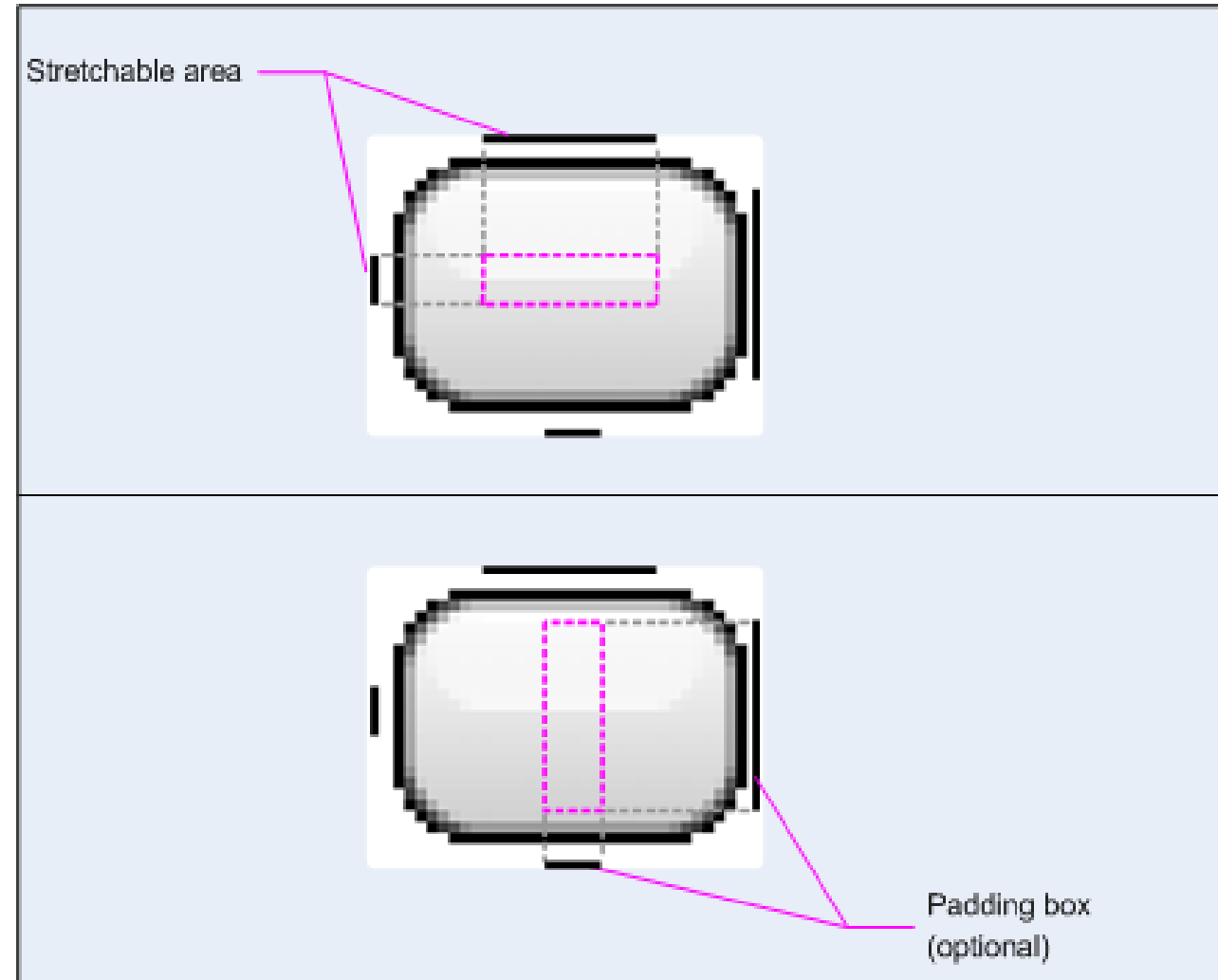
- Avoid hard-coded layout sizes
  - Use "wrap\_content" and "match\_parent" instead

```
<TextView  
    android:layout_width="match_parent"  
    android:layout_height="wrap_content"  
    android:text="@string/lorem_ipsum" />
```



# Create stretchable nine-patch bitmaps

- A nine-patch bitmap is basically a standard PNG file, but with an extra 1px border that indicates which pixels should be stretched
- Optionally, we can define the safe region where content should go inside the view by similarly adding lines on the right and bottom edges



# Network-Aware Adaptation

# Network Changes in Mobiles

- Mobile users are often in an environment with multiple data communication networks available.
- Users on the move may come across variations in networking conditions



VectorStock®  
VectorStock.com/22963714



# Adaptation based on bandwidth

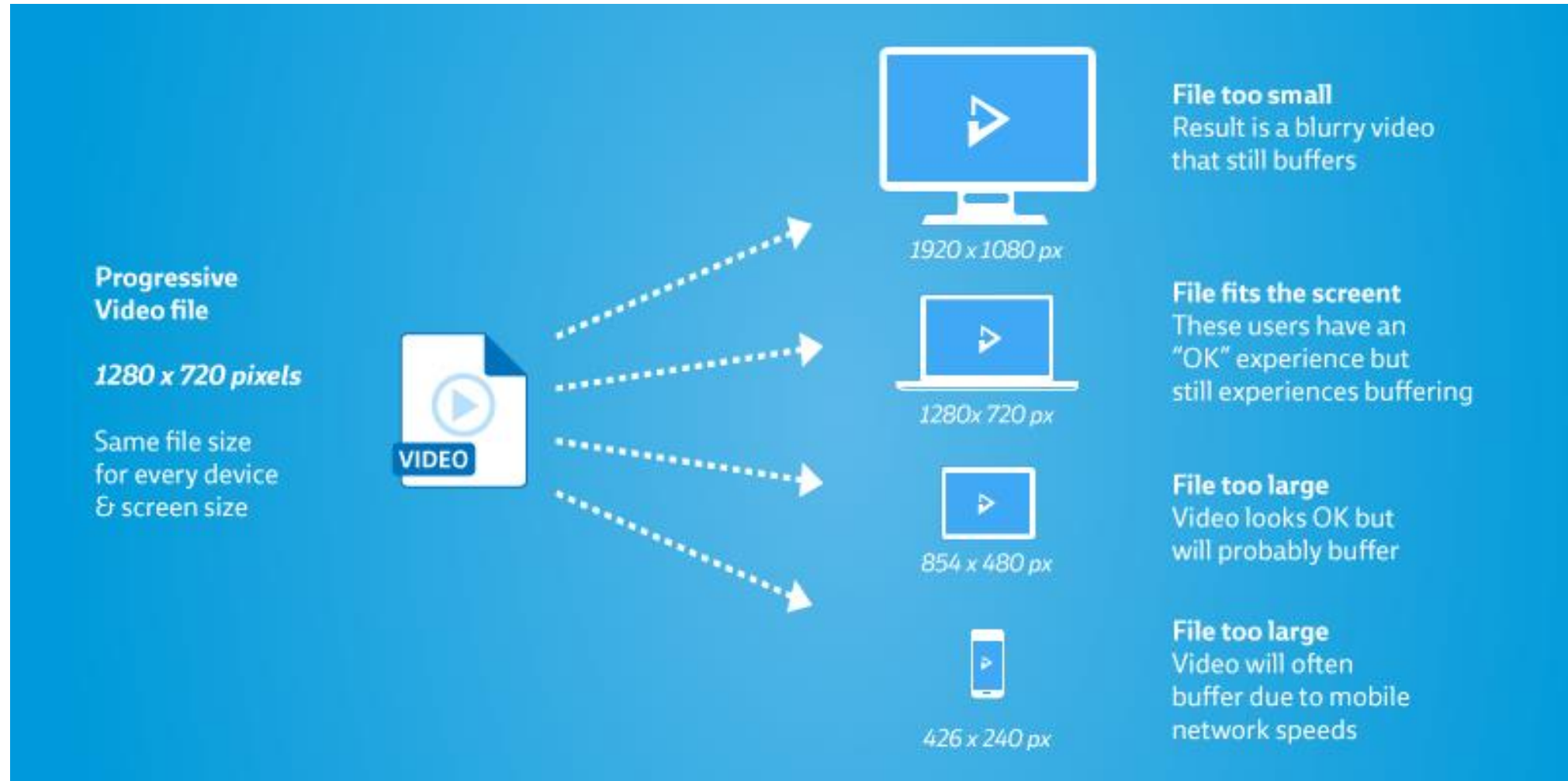
- The content can be adapted to a lower fidelity when a lower bandwidth link is detected.
  - For example, use the round trip time to optimize the image quality
  - Content can be adapted by using degradation mechanisms to reduce the quality of the content.
  - Opportunistically procure extra resources in order to maintain the quality of the content

# Adaptive Video Streaming

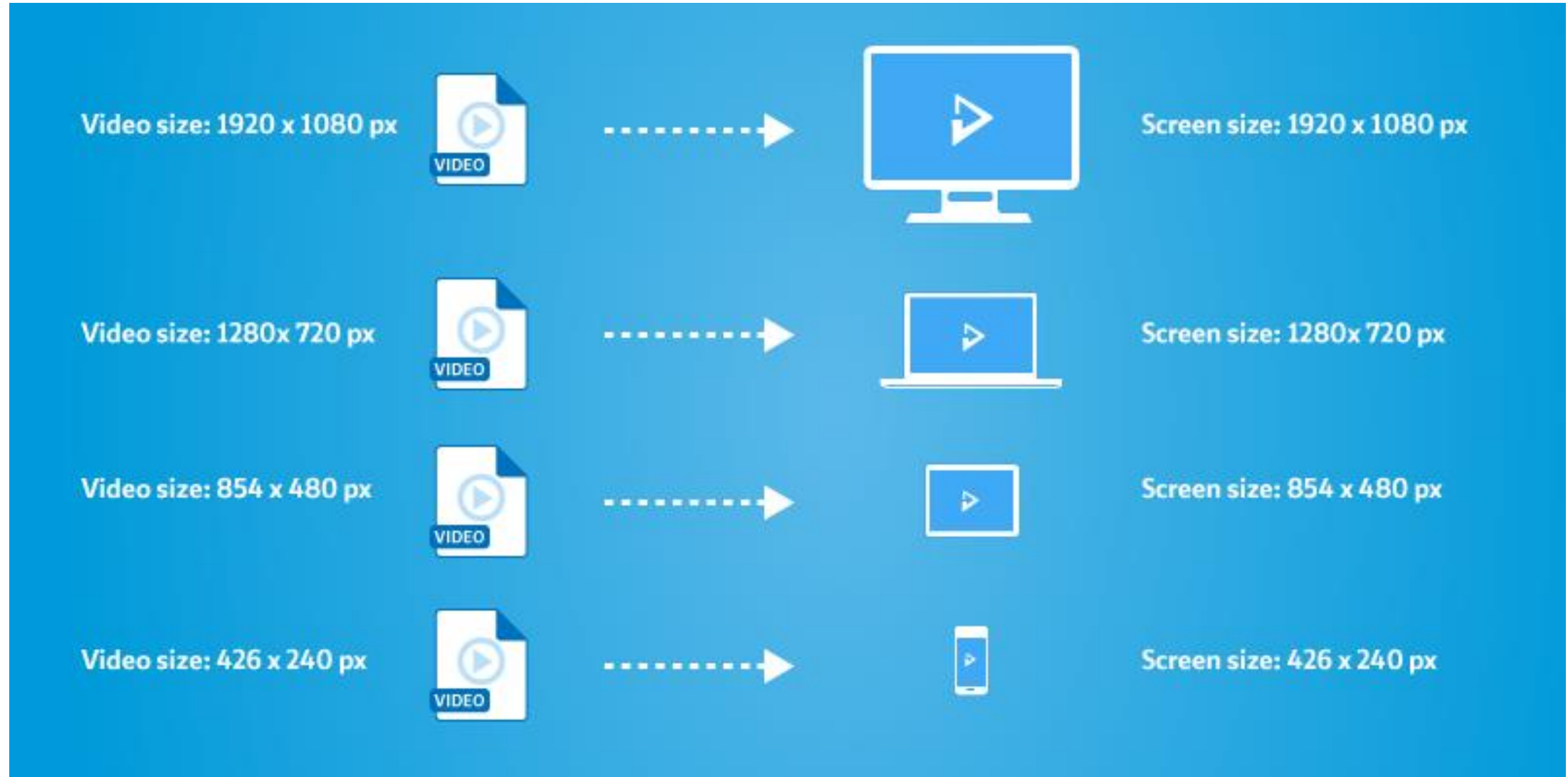
- E.g., Dynamic Adaptive Streaming over HTTP (DASH) in Youtube
- Motivation: one size does not fit all



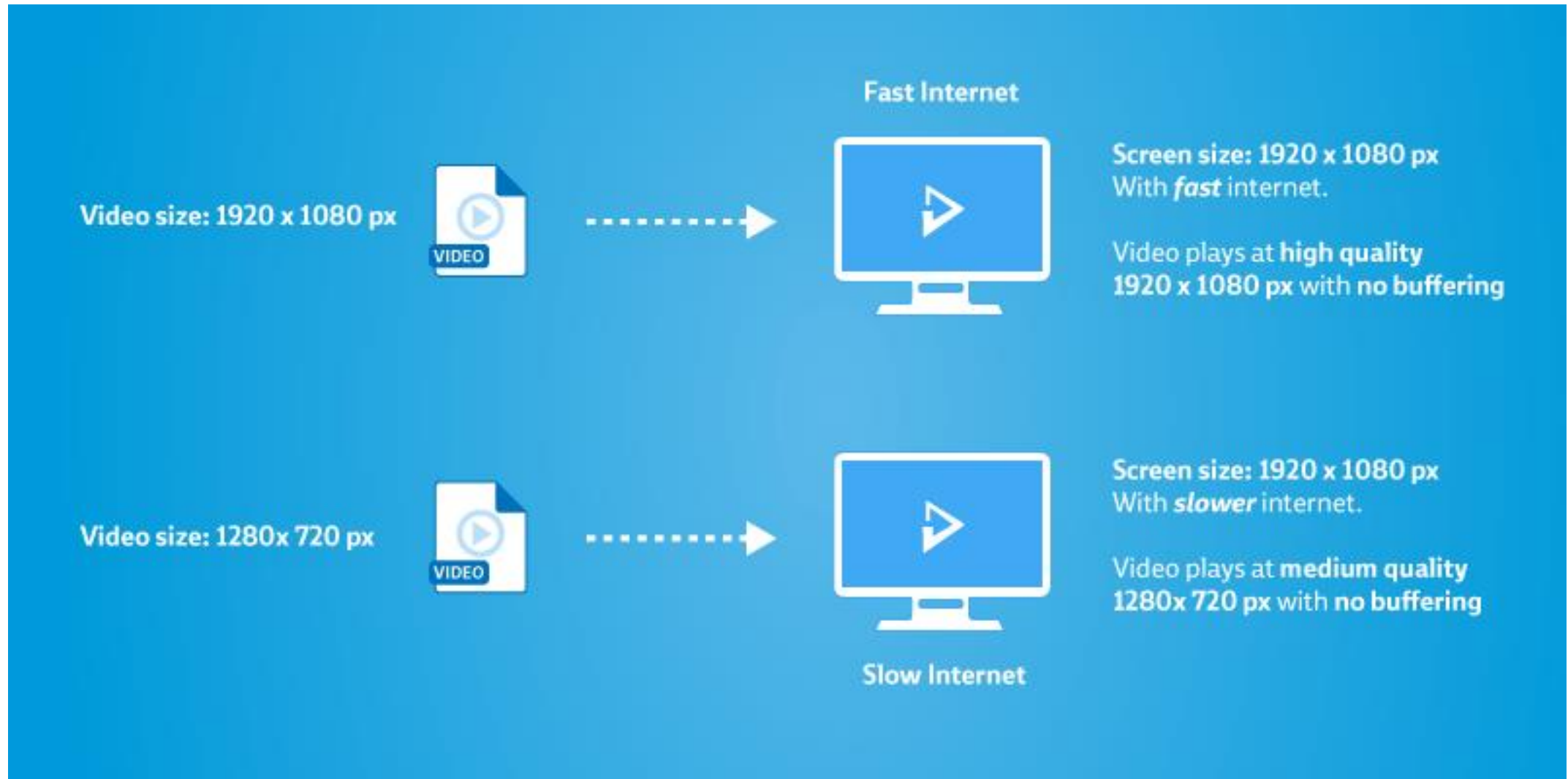
- Baseline: a progressive video stream is simply one single video file being streamed over the internet, then stretch to different screens.



- Adaptive streaming: allows the video provider to create a different video for each of the screen sizes



- Adjust the video size based on network connection quality



- The biggest strength: adaptive bitrate

