SNS COLLEGE OF ENGINEERING

Kurumbapalayam (Po), Coimbatore – 641 107

An Autonomous Institution

Approved by AICTE, New Delhi & Affiliated to Anna University, Chennai

DEPARTMENT OF COMPUTER SCIENCE AND TECHNOLOGY

COURSE NAME: 19CS603-Mobile Application Development

III YEAR /VI SEMESTER

UNIT – II Basic Design

Design constraints for Mobile applications - Both Hardware and Software related

1. Screen size, sensors, and interactions

Screen sizes are much smaller when designing mobile software. You have a limited canvas,and your design should be simple with enough space for users to touch and interact with different elements. Users may use your software with one hand or two hands or use any of the supported gestures to interact with your software.

Depending on your content, you could choose a spatial format (map view), a list format, a block design, or other ways to display your content. If you are building for iOS or Windows Phone, there are fixed screen sizes and resolutions to plan for. If you're building for Android, you have more variations to keep in mind.

You also have a variety of sensors and enablers that can help you design interactions. While many of them are great enablers for design, they also come with their constraints (eg: GPS when used indoors with a spotty data connection may not return a location. How will your software handle this?)

2. Storage and cache sizes

Depending on what you're building, you might have options to download and store/cache content for offline usage. This could help reduce the data transfer for online transactions, making the product feel more responsive. Think about whether the storage is available on a memory card (removable storage) or internal memory. With a memory card, you must tackle states where the user might delete/modify content on the memory card, pull out the memory card while interacting with the software or for memory card corruption. With internal software, you may have limited allotment for storage, and the possibility of cached data getting overwritten.

3. Latencies

It's a good idea to time operations and load times. Users on a mobile expect an instant response for something they do.

A lot of this depends on your product architecture. You may be able to do some architectural jugglery to deliver a phased experience. Think of how images load in your browser on a slow connection – a lighter pixelated version loads first, and then the sharper image loads. The intermediate stage keeps the user occupied till the final image loads.

Even if you can't eliminate all latencies, you could plan to tackle the intermediate wait stages with engaging messages or cached content. First time use is a special case. When the app is used for the very first time, there are usually a lot of background setup activities that need to be done. One way of mitigating the wait during this time is to have a front-end tutorial for users to engage with, while your app performs setup or bookkeeping in the background. On subsequent usage, you could use data/states cached to deliver a faster experience.

4. Network issues

Any mobile product must contend with network latencies and failure points. Plan for these earlier on, or you may find you're losing users because they keep staring at a 'Try again later' or 'Connecting…' message. Try to avoid blocking spinners – one that does not let the user do anything while he waits. See if your platform and design permit loading partial elements and cached data to keep him occupied when he waits.

Depending on the platform you develop for, you may also have cases where the user can pull out the SIM card during an operation and put it back in or walk in and out of Wi-Fi zones. This affects not just the software on the phone, but also the backend that supports it. Finally: try identifying failure points and use that as avenues to show your product's personality. Twitter's fail-whale is a great example of how a failure point can become an icon.

5. Data use requirements

In addition to just data connection, the SIM card also provides information like MCC (Mobile Country Code) and MNC (Mobile Network Code) that you may have used to identify which country the user is and what network he is connected to. You can get info on whether he is on a roaming network and home network and adjust the amount of data required accordingly. For example, you may choose to download and cache a lot more data when the user is connected to Wi-Fi or a home network but take a download-as-required approach when he is on roaming.

6. Fonts, language, and tone of voice

The fonts and language you use in the product makes a big difference to the user experience. Language and tone of voice reflect your product's personality. The tone can be informal (Howdy!), formal (Welcome), friendly (let's get started), impersonal (press this button to continue) or any other variation that suits your product. It's important to realize what tone your customers are comfortable with and stick to one tone through the product.

Point to note: if you're building a global product, be careful about the tone of voice you choose. What comes across as informal in one culture may seem blasé in another.

Languages may bring their own issues – getting translations, checking that terms do not get

curtailed, etc. You may also have to plan for right-to-left languages like Arabic.

7. Corner cases for product usage

There are many use cases that you may not be able to test during development. For example, it might be difficult for you to test ‘international roaming’ use cases unless you procure an international SIM or send an Indian SIM to a tester/friend sitting abroad.

S.VIJAYALAKSHMI, AP/CST,SNSCE