Porting Application Code

CSC 4700 Software Engineering

What is Porting?

- Porting is the ability to reuse software from one environment to another by making minimal changes to the existing code.
- Unfortunately, these minimal changes are not easy to do when it comes to porting software from one Operating System (OS) platform to another or even among versions of a single OS
- If porting requires lot of changes to the existing code, then it's called a "code re-write"

Why Port?

- Developers need to leverage existing software as opposed to wasting the software investment and starting from scratch

Importance of Porting

- Porting is usually quicker and cheaper than performing a full code rewrite.
- However, manual porting can turn into a major code-rewrite if the underlying OS platforms are very different
- Ideally, If you can re-use your existing code without having to do any “porting” at all, then it’s all the more better

Importance of Porting

- Existing code works and has been perfected in the field.
- Throwing it away and starting from scratch because of a change in environments is not sensible
- Porting is more effective and efficient if there is a lot of code

Challenges in Application Porting

- Differences in OS versions
- OS differences between 32 and 64 bit CPU
- Missing OS functionalities
- Discontinue use of obsolete APIs
- Change code to adapt to new OS versions
- Manage multiple code bases across different versions
- Forced to upgrade OS due to vendor not supporting a specific OS version. As a result, also forced to upgrade hardware.

Challenges in Application Porting

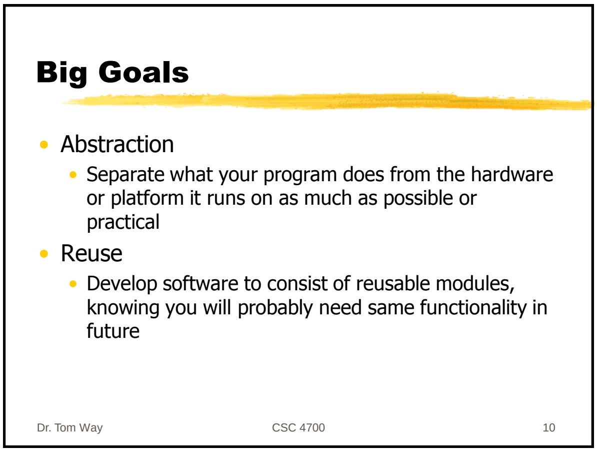
- Code may compile on one compiler but not another
- Code developed in a specific language may become obsolete (Ada, Fortran, COBOL, etc.)
- Tools not available for new hardware platform
- Development on host may not be possible
- Delay in start of development until the new target hardware/tools become available
- Need to deal with target issues while porting

Challenges in Application Porting

- Longer-term porting solution:
  - Develop common OS interface APIs that your code can reuse across multiple OS & their versions, otherwise called an in-house OS abstraction
  - Check out the commercial OS abstraction solutions
  - Engage a porting consultant

Porting Tools

- Make it easy to support a wide variety of OSs
- Provide tools that can run in simulation and debugging modes
- Deliver tested and proven architecture
- Provide postmortem analysis of the code with tools
- You may have to provide the source code to a third party if you utilize them to do the port. Do it yourself by using a tool
- Complete porting in less time
  - Shorten your time to market



Big Goals

- Abstraction
  - Separate what your program does from the hardware or platform it runs on as much as possible or practical
- Reuse
  - Develop software to consist of reusable modules, knowing you will probably need same functionality in future