CS 430 - Lecture 20 - Portability

Collin Roberts

November 20, 2023

Outline

Portability Definition

Definition 1

A program, P1, is **portable** *if it is* **significantly** *cheaper to convert it to P2 (and run it on H/W H2, with OS O2 & compiler C2) than to re-code P2 from scratch.*

Portability Definition

Remarks:

1. Portability does not mean porting the code only:
   1. We must port documentation & manuals too.
   2. If S/W is changed, then all docs must also change.

Hardware Incompatibilities

1. Character codes:
    1. American Standard Code for Information Interchange (ASCII): 00000001
    2. Extended Binary Coded Decimal Interchange Code (EBCDIC): 10000001
    3. S/W developed on a platform with one encoding must be modified to work on a platform with the other encoding.

Operating System Incompatibilities

1. MAC OS versus Windows.
2. Similar problems on mainframe-scale systems.
3. JCL (**J**ob **C**ontrol **L**anguage, for specifying all the parameters needed to run mainfrmame batch jobs)
    1. Each OS's JCL is slightly different.

Operating System Incompatibilities

4. Virtual Memory (i.e. augmenting physical memory by allocating some disk space as virtual memory)
    1. If S/W is developed on an O/S that supports virtual memory, then there is no practical limit on the amount of memory available.
    2. But if that same S/W is then ported to an O/S that does not support virtual memory, then there is a hard limit on the amount of memory available.

Numerical System Incompatibilities

1. Word size:
    1. S/W developed on a 64-bit platform will not run on a 32-bit platform.

Compiler Incompatibilities

1. Different compiler versions can enforce different syntax rules.
    1. Often newer compilers are more strict.

Is Portability Really Necessary?

Q: Does it make sense to spend time/resources to develop portable S/W?

A: Yes:

1. If your firm's business is selling software, then portability = higher revenue.
2. Even if not, i.e. if your organization builds software to support another primary business (e.g. selling insurance at SLF), keep in mind that good software lives 10-20 years or more, while hardware changes every 4-5 years. So portability saves money here too.

UNIX

1. UNIX O/S was constructed for maximum portability:
   1. platform-independent (portable):
      1. 9000 LOC written in C
   2. platform-dependent (must be re-written for each platform):
      1. 1000 LOC written in Assembly
      2. 1000 LOC of C - device drivers

UNIX

2. Lessons of UNIX
    1. We should emulate the techniques used to design/build UNIX as much as possible.
    2. When we have a choice of O/S, we should choose UNIX.

Portable Application Software

1. Although we may not always have control over which programming language we must use, whenever possible we should choose a high-level language (higher-level = more insulated from the hardware level).

Portable Data

Porting large amounts of data can be very problematic.

1. Flat files are the most portable data format. Problems:
   1. Misunderstandings about file formats.
   2. Self-documenting file formats (e.g. XML) solve problem 1, but make files get big.

Object-Oriented Technologies (OOT)

1. Major promise of OOT:
   1. final S/W product is portable & reusable