Welcome to CS 241
Systems Programming at Illinois

Wade Fagen

The Team

- Wade Fagen
  - Office: 4101 SC
  - cs241help-su12@cs.illinois.edu
- Lab Assistants
  - Yang Xu
  - Brian Wang
- Discussion Sections
  - 2 sessions (T/W 2p-3p or T/W 3p-4p)
  - All sections in SC 0220

News and Email

- Announcements and discussions: Piazza
  - http://www.piazza.com/illinois/cs241
    - All class questions
    - This is your one-stop help-line!
    - Will be checked daily by staff
- E-mail
  - cs241help-su12@cs.uiuc.edu
  - Regrades, personal questions, etc
    - Will be slower response-time than Piazza

The Textbook

- Introduction to Systems Concepts and Systems Programming
  - University of Illinois Custom Edition
  - Copyright © 2007
  - Pearson Custom Publishing
  - ISBN 0-536-48928-9

- Taken from:
  - Operating Systems: Internals and Design Principles, Fifth Edition, by William Stallings
  - UNIX™ Systems Programming: Communication, Concurrency, and Threads, by Kay A. Robbins and Steven Robbins
  - Computer Systems: A Programmer's Perspective, by Randal E. Bryant and David R. O'Hallaron

Your CS 241 "Mission"

- Come to class
  - MTW, 11-12:50am
  - Participate actively
  - Attend 1 discussion section per week
- Read textbook
  - Reading assignments posted on webpage
- Homework (1) 3%
- Programming assignments (8) 47%
  - One MP /week; 5% for MP1, 6% for MP2-8
- Midterm 20%
  - Monday, July 9th, 11am – 12:50pm (In Class)
- Final 30%
  - Saturday, August 4th, 1:00pm – 3:00pm

It's all about the programming!

- MPs
  - Goal
    - Expose you to the concepts and APIs taught in class
  - All individual
    - You can't learn it if you don't do it yourself!
- MP Contest
  - Memory (**`malloc`**)
- Components for grading
  - Correctness
    - Autograder
    - Once a night to help you check correctness
    - Does not reflect grade
  - Memory
    - Correct usage?
    - All memory free?
    - valgrind

Deadlines

- Homework
  - Deadlines are strict
  - Late submissions will not be considered
- MPs
  - Please respect posted deadlines to ensure quick grading
  - Late MPs will be penalized 2% for each late hour (rounded off to the higher hour)
  - No submissions past 24 hours

Regrades

- Within one week of posting of grades for a quiz, homework, MP or exam
- Regrades must be submitted in writing on a separate piece of paper
  - Please do not write on your homework, MP or Exam

Academic Honesty

- Your work in this class **must** be your own.
- If students are found to have collaborated excessively or to have blatantly cheated (e.g., by copying or sharing answers during an examination or sharing code for the project), **all** involved will at a minimum receive grades of 0 for the first infraction, lose a grade letter, and reported to the academic office.
- Further infractions will result in failure in the course and/or recommendation for dismissal from the university.
- Department honor code: https://wiki.engr.illinois.edu/display/undergradProg/Honor+Code

What is cheating in a programming class?

- At a minimum
  - Copying code
  - Copying pseudo-code
  - Copying flow charts
- Consider
  - Did some one else tell you how to do it?
- Does this mean I can't help my friend?
  - No, but don't solve their problems for them

Course Questions

- What is an operating system?
- What is it for?
- How do I use it?
- What is concurrency?
- What is system programming?

This is the name of the class – but there is a lot more to 241 than just programming!

Course Objectives

- By the end of this course, you should know about operating systems
  - Identify the basic components of an operating system
  - Describe their purpose
  - Explain how they function
- Use the system effectively
  - Write, compile, debug, and execute C programs
  - Correctly use system interfaces provided by UNIX (or a UNIX-like operating system)

General Course Outline

- Week 1: Basics of Systems / C Programming
- Week 2: Memory
  - Heap allocation, paging, virtual memory, fragmentation
- Week 3: Processes / Threads
  - Process/thread isolation, pthread library, mulit-thread programming
- Week 4: Scheduling / Synchronization Introduction
  - Scheduling strategies/analysis, deadlock, starvation, classical problems
- Week 5: Synchronization
  - Deadlock detection/avoidance/prevention, mutexes, semaphores, cond. vars
- Week 6: IPC
  - Signals, pipes, FIFO, shared memory, I/O multiplexing
- Week 7: Networking
- Week 8: File Systems, I/O, and beyond CS 241

Complete Schedule

- See class webpage
- http://www.cs.illinois.edu/class/cs241
  - Schedule is dynamic
  - Check regularly for updates
- Content
  - Slides will be posted before class
    - Some class material may not be in slides
      - Examples may be worked out in class

Your to-do List

- Visit the class webpage
  - Check out all the info
    - Especially schedule, grading policy, homework & MP hand-in instructions, and resources
  - http://www.cs.illinois.edu/class/cs241
- Familiarize yourself with Piazza
  - See http://piazza.com/
  - Access Code: _____
- Find a reference to refresh your C programming skills
  - http://www.lysator.liu.se/c/bwk-tutor.html

What is systems programming?

sys·tem **Noun /ˈsistəm/**

1. A set of connected things or parts forming a larger and more complex whole.
2. An integrated set of elements that accomplish a defined objective

- Examples: Digestive system, economic system, ecosystem, social systems

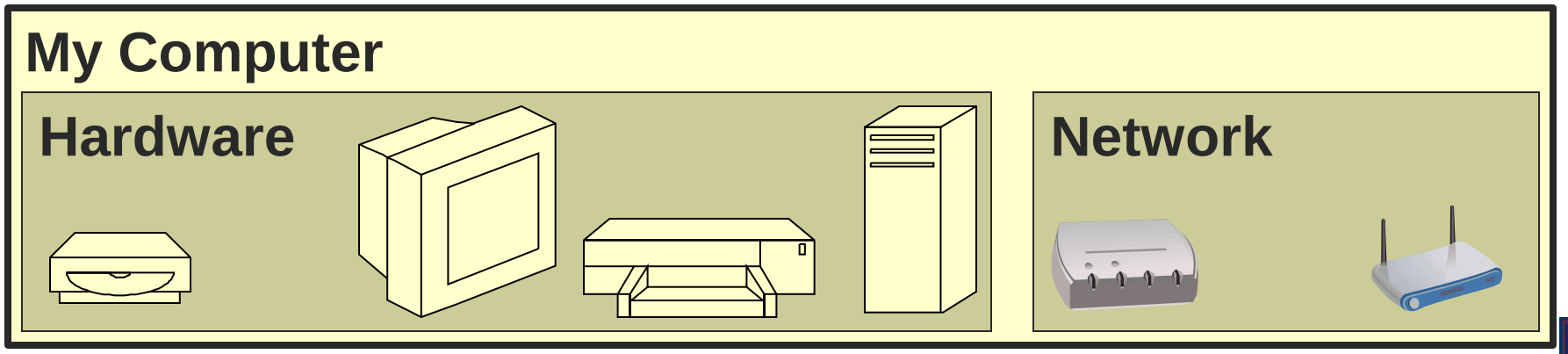
- Computer systems: collections of programs
  - Search engines, social networks, databases, Internet
  - In this class, we learn how to design and implement computer systems

Challenges in programming computer systems

- Making programs share resources
- Preventing malicious/incorrect programs from interfering with other programs
- Coordinating operations of multiple programs
- Communicating information between programs

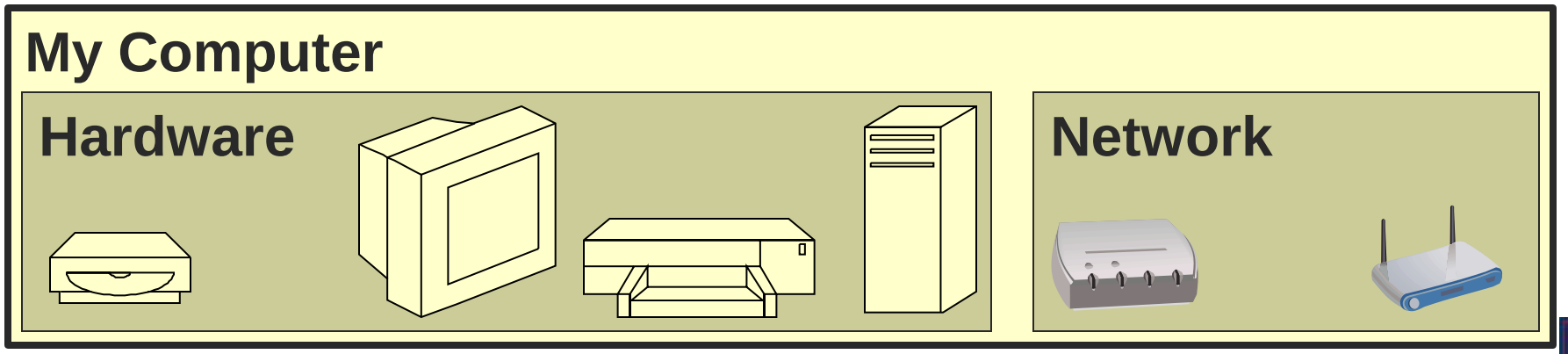
What is an operating system and why do I need one?

- What do we have?
  - Set of common resources



What is an operating system and why do I need one?

- What do we have?
  - Set of common resources
- What do we need?



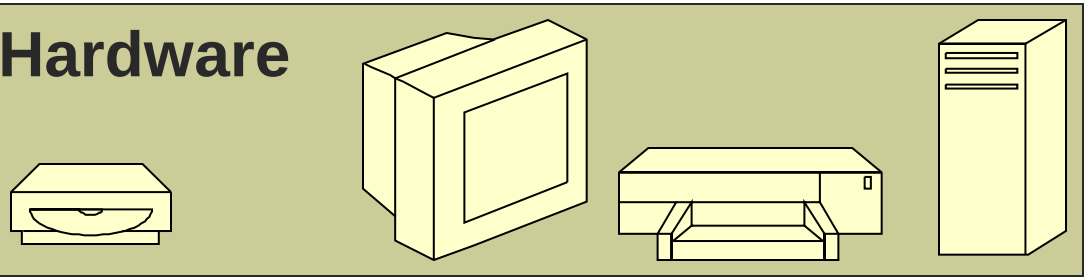
What is an operating system and why do I need one?

Application Software

Firefox | MS Word | Diablo III | SSH

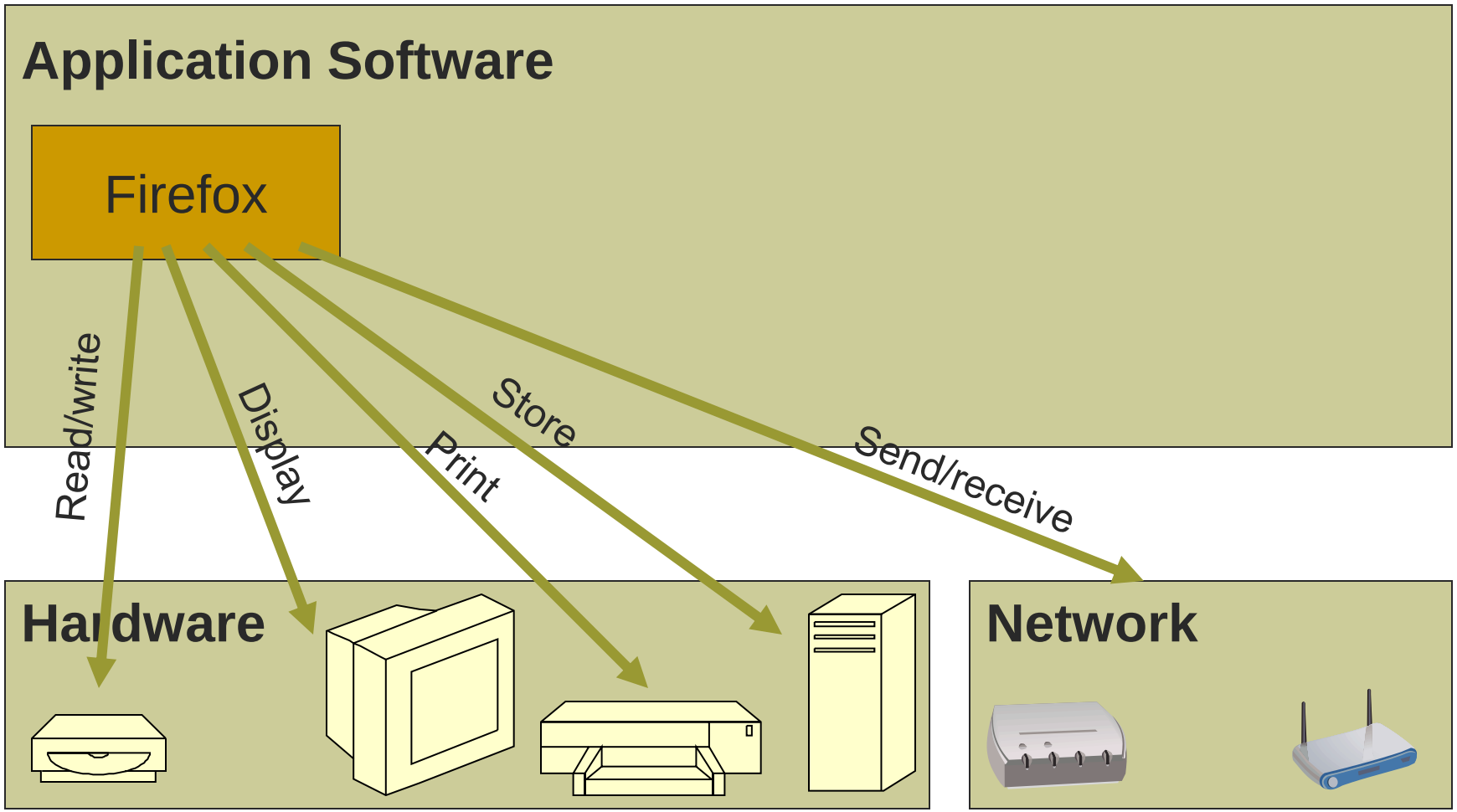
- A clean way to allow applications to use these resources!

Hardware

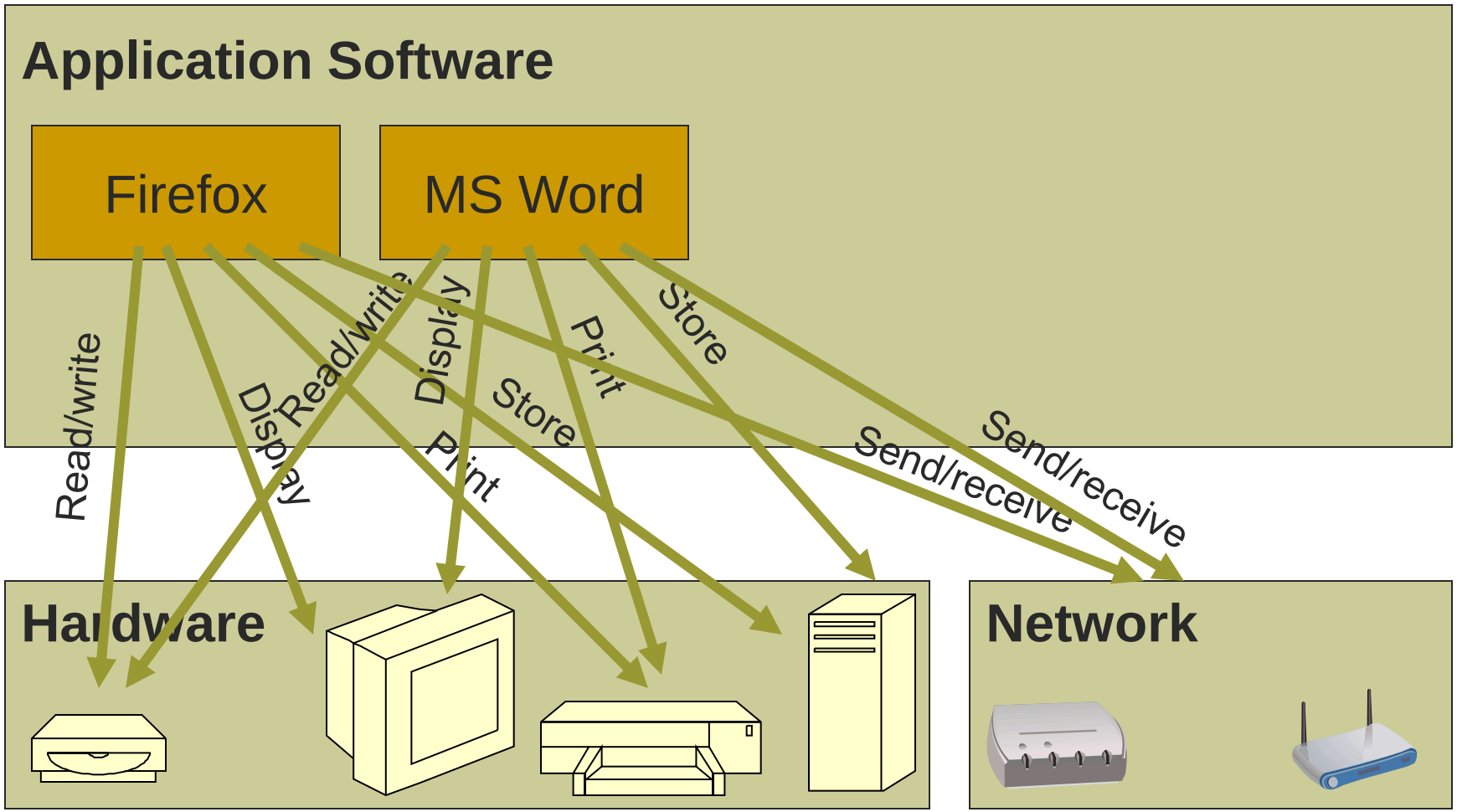


Network

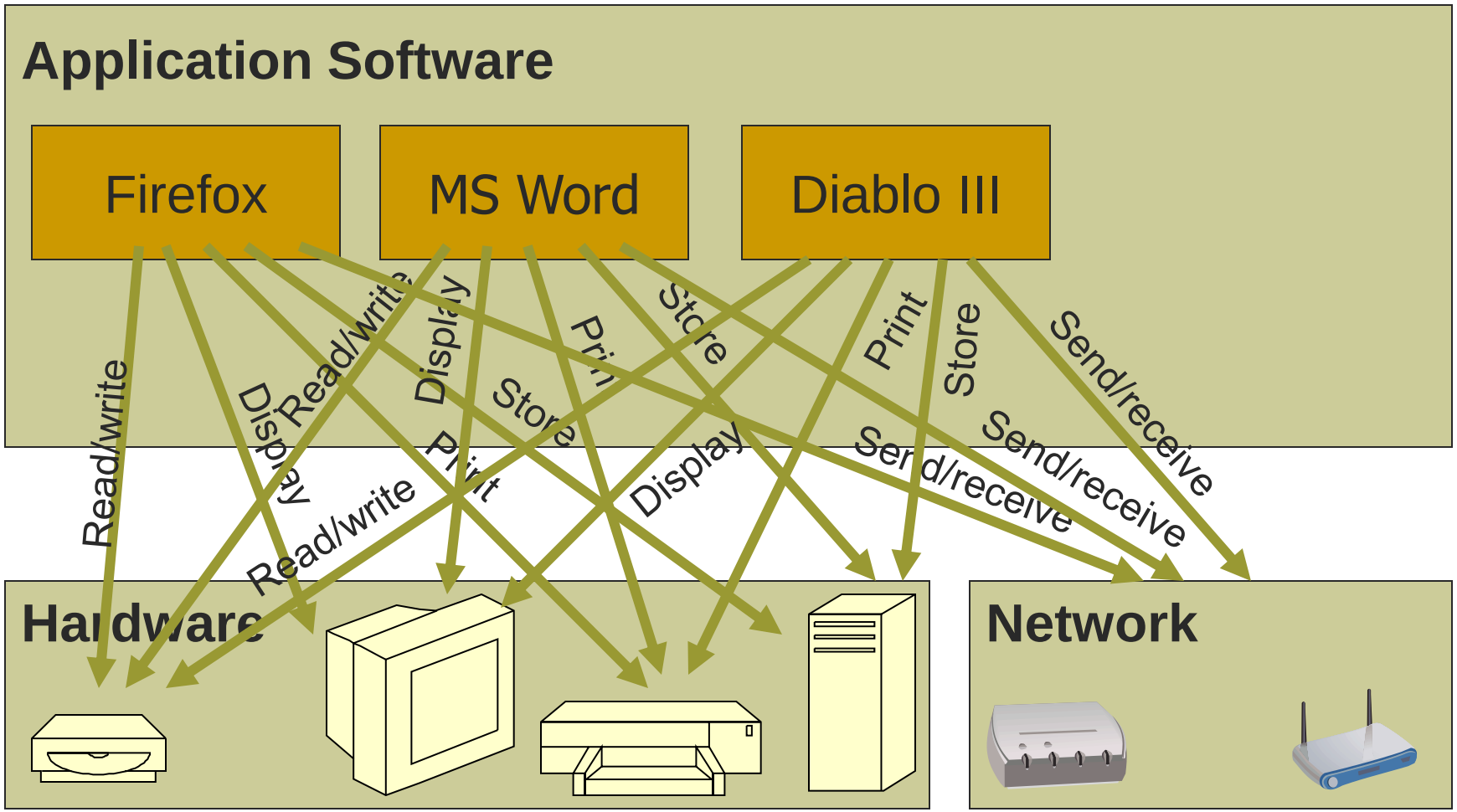
Application Requirements



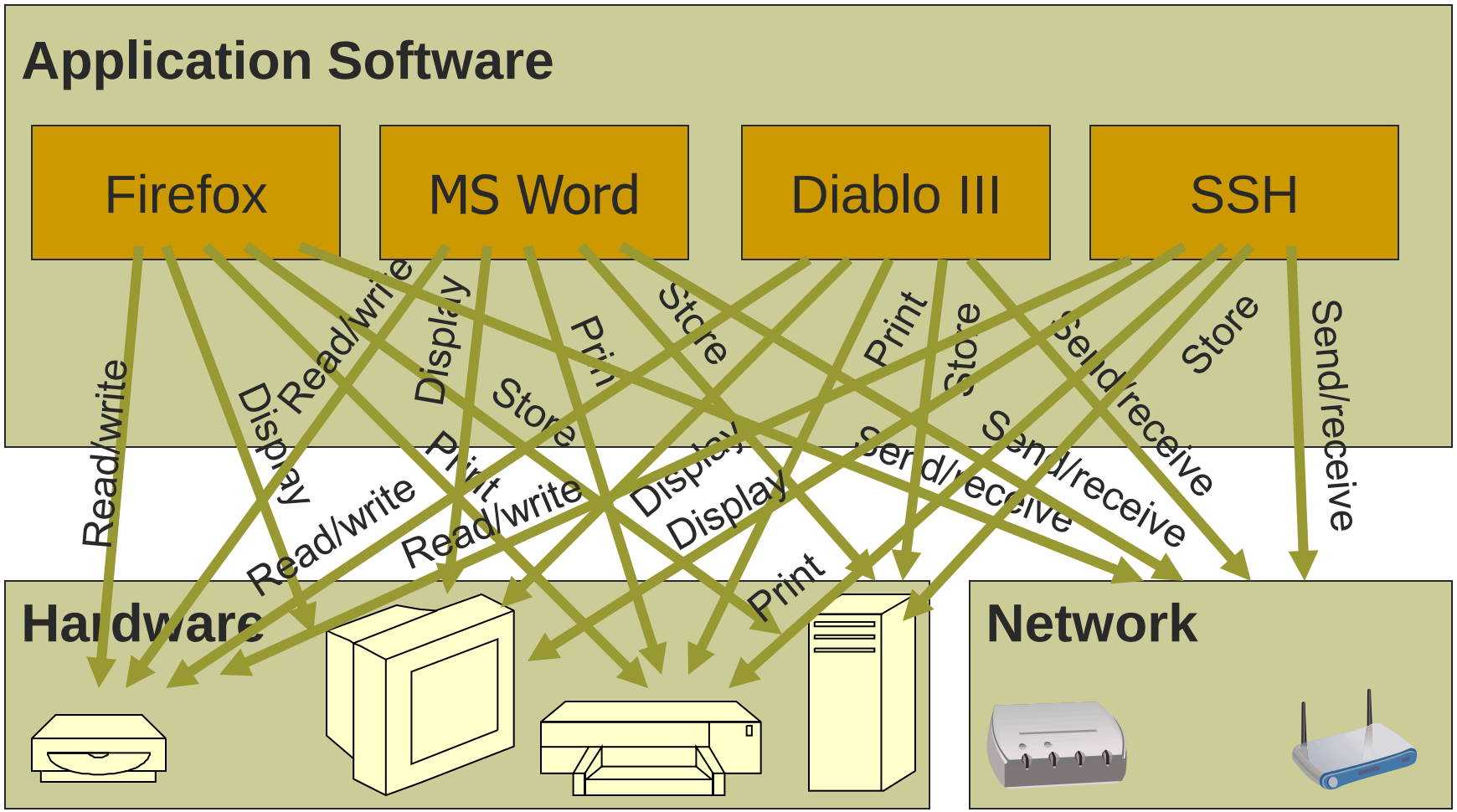
Two Applications?



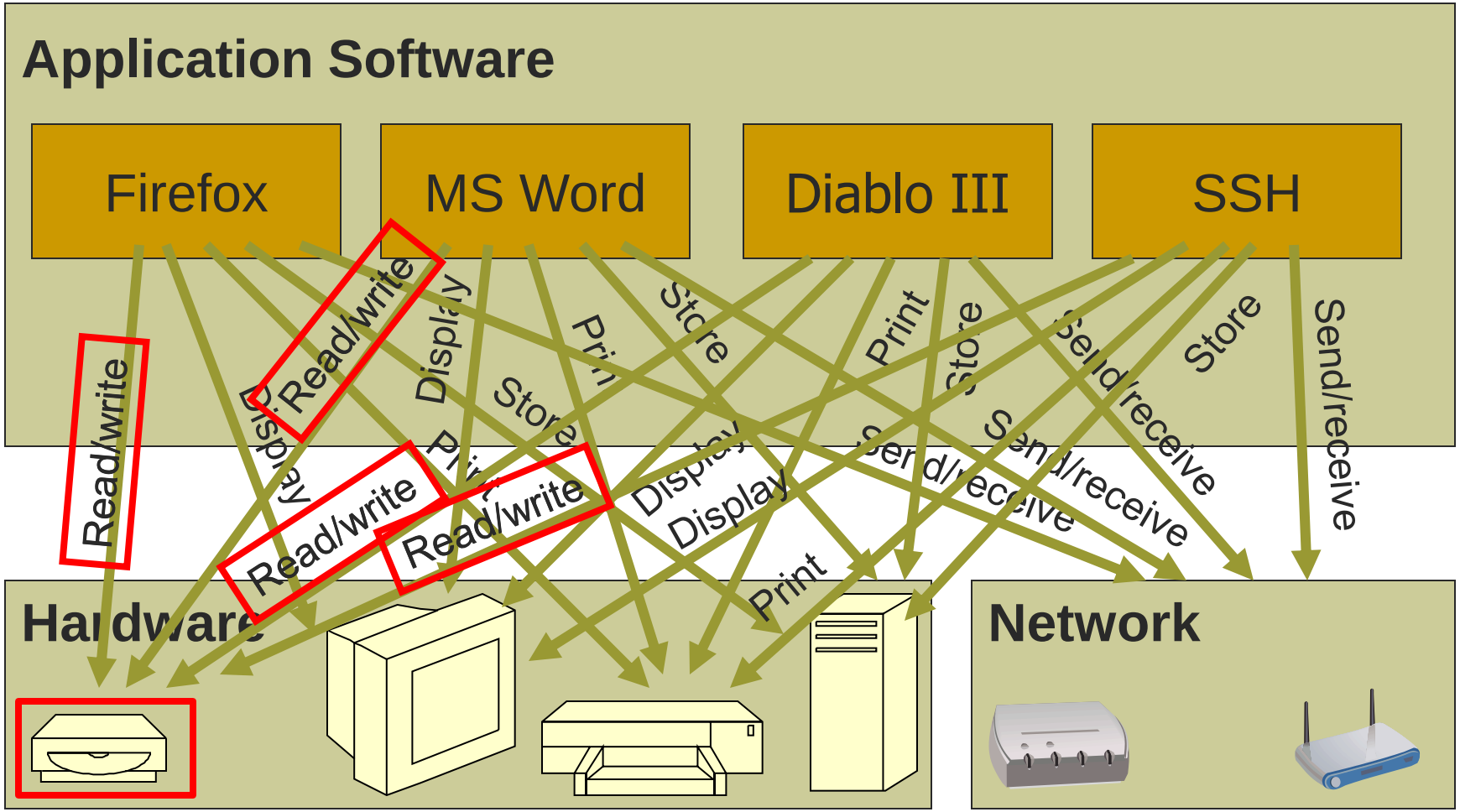
Managing More Applications?



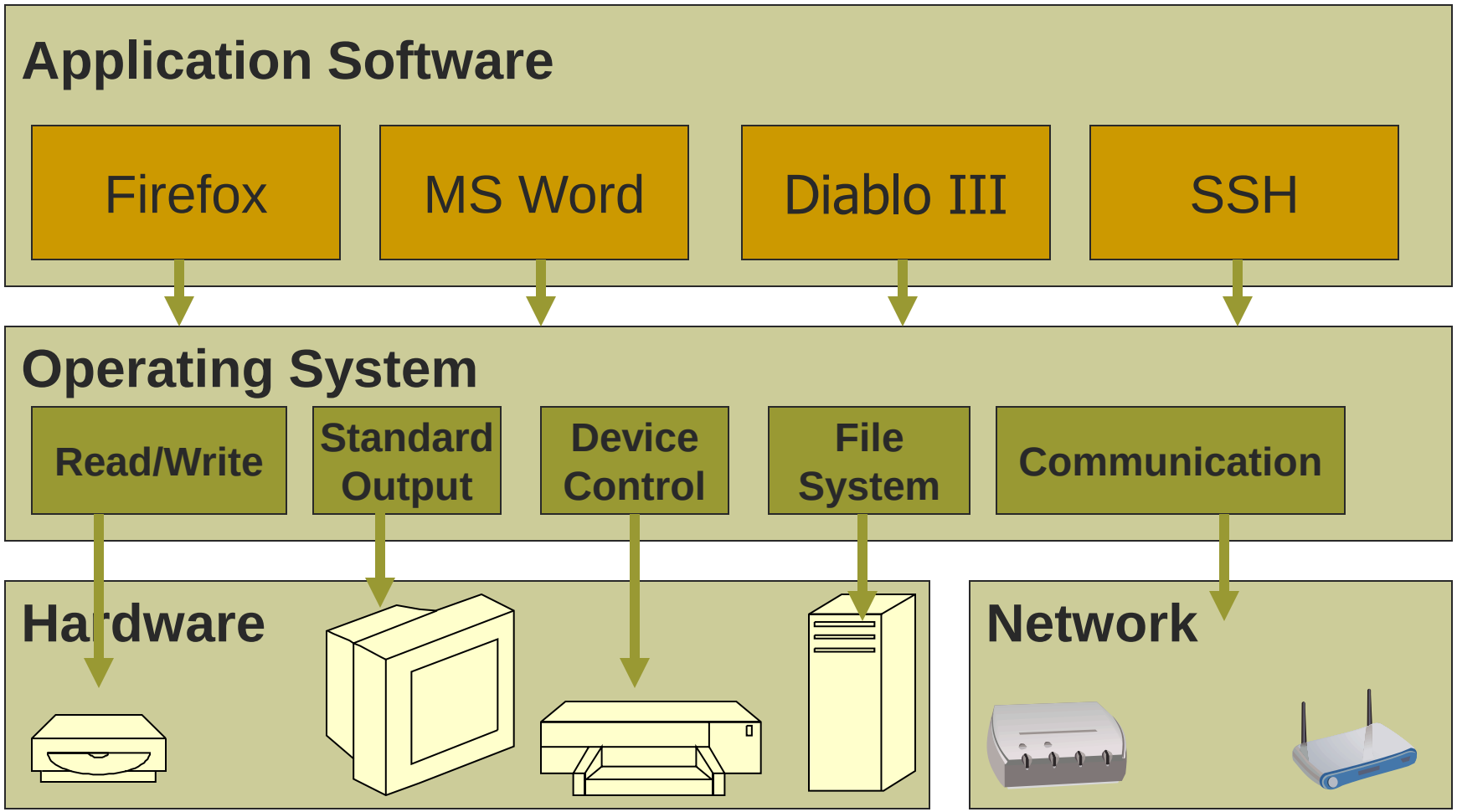
We need help!



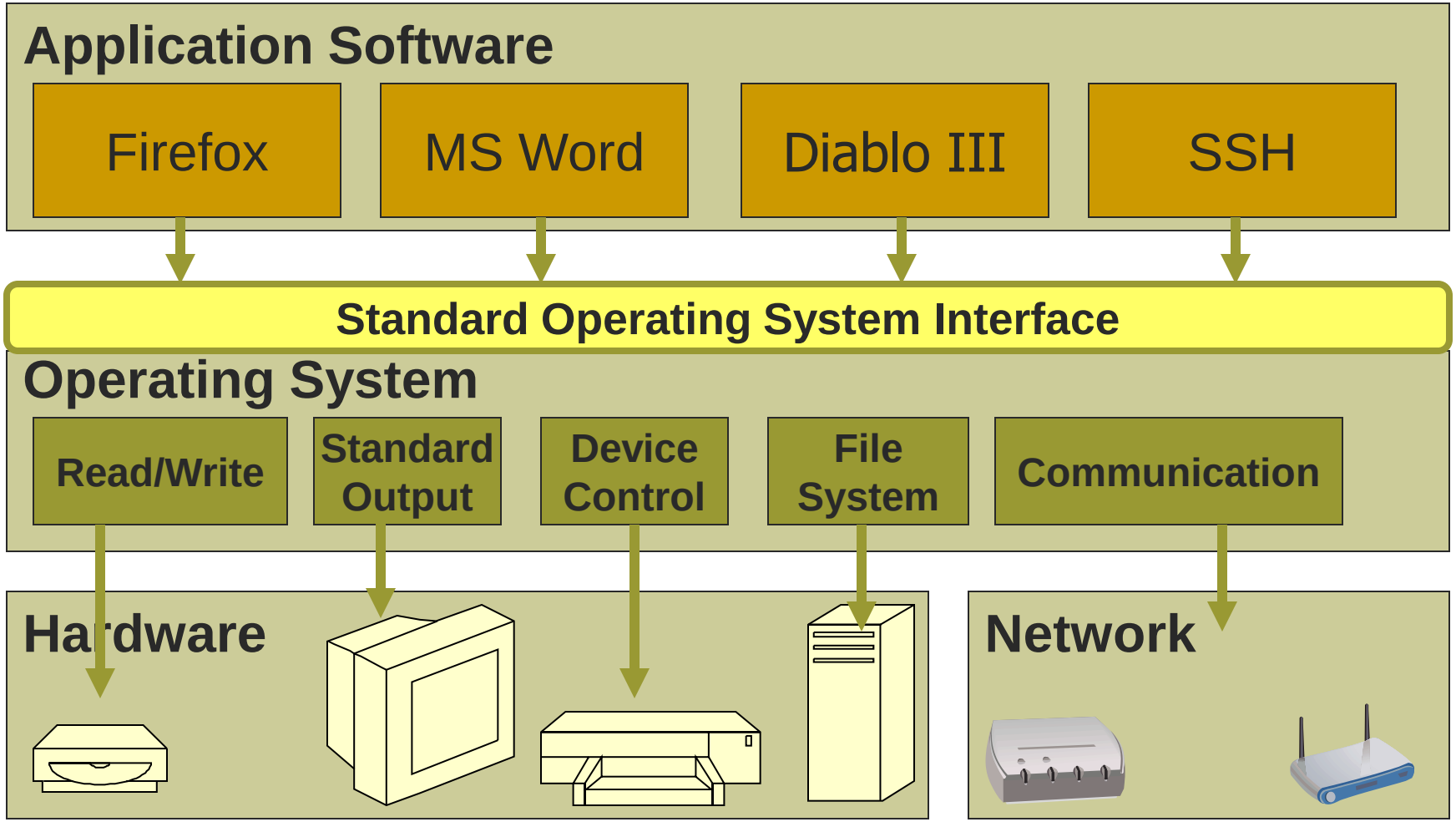
Approach: Find Common Functions



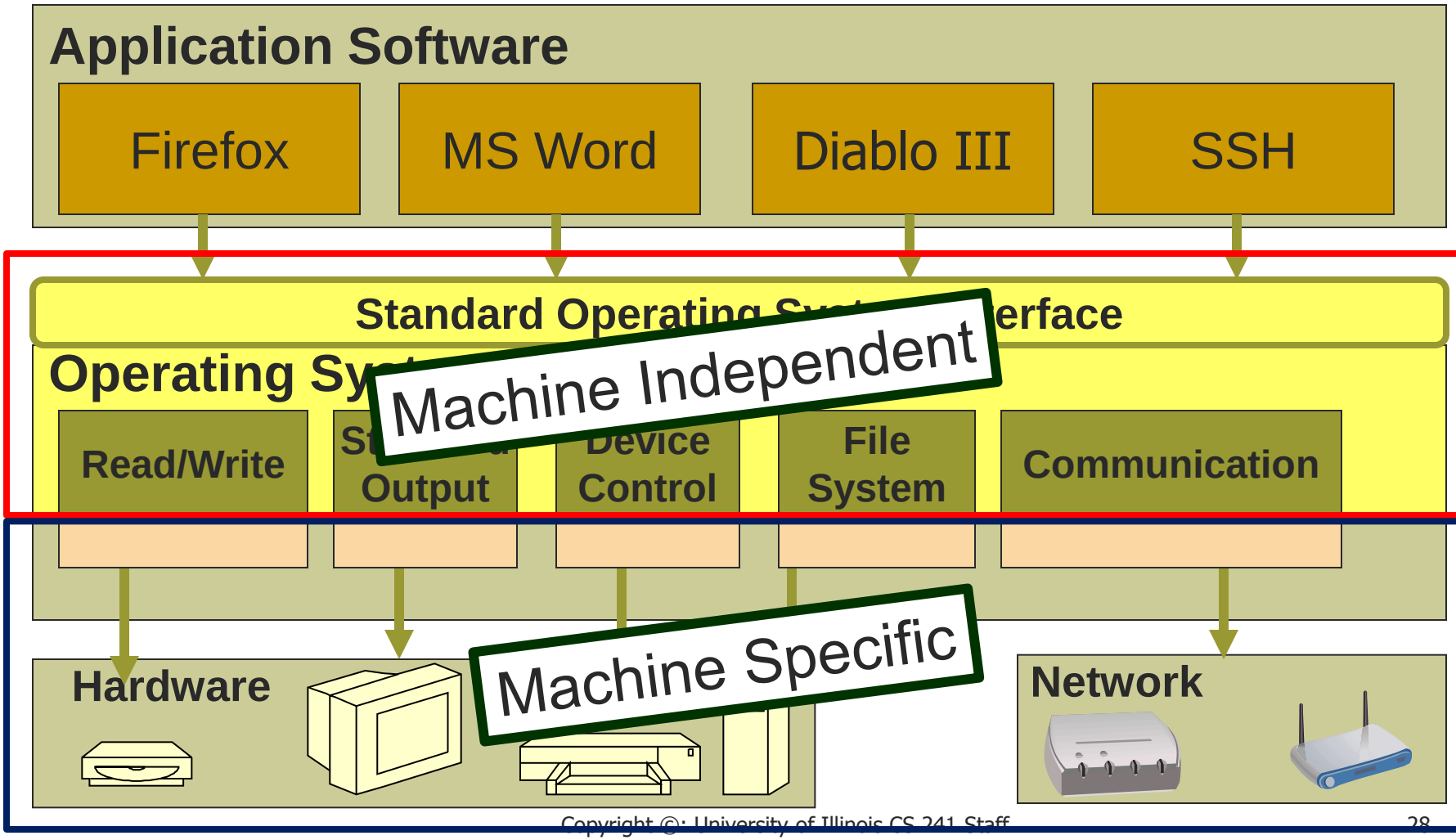
Delegate Common Functions



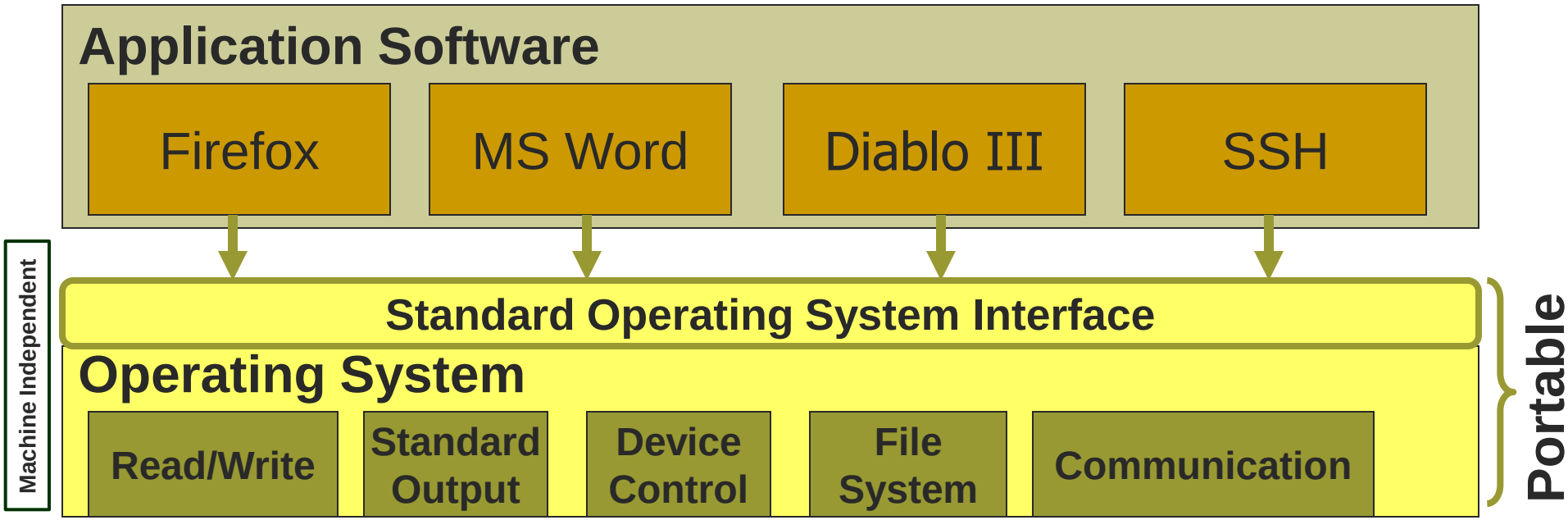
Export a Standard Interface



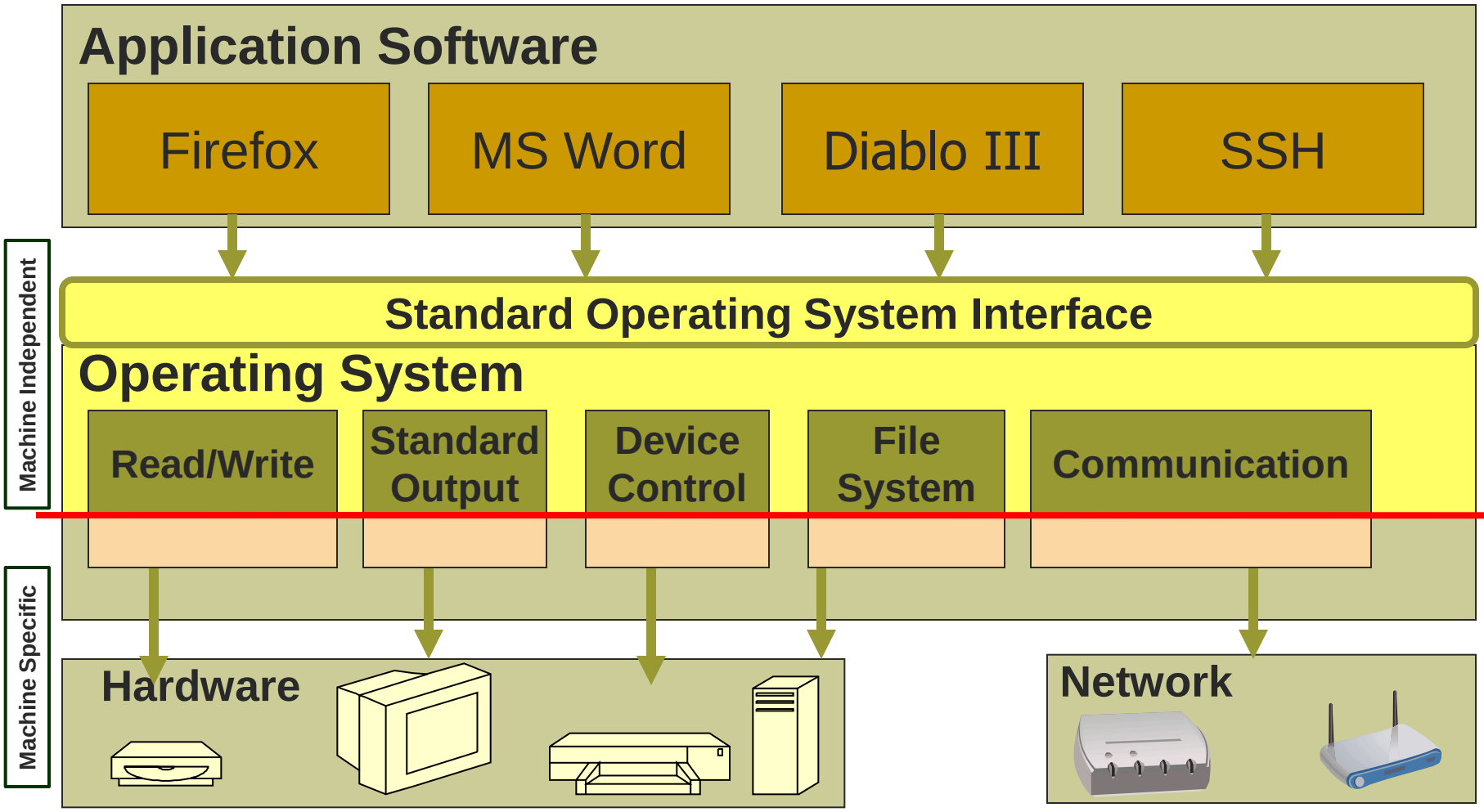
Goal: Increase Portability



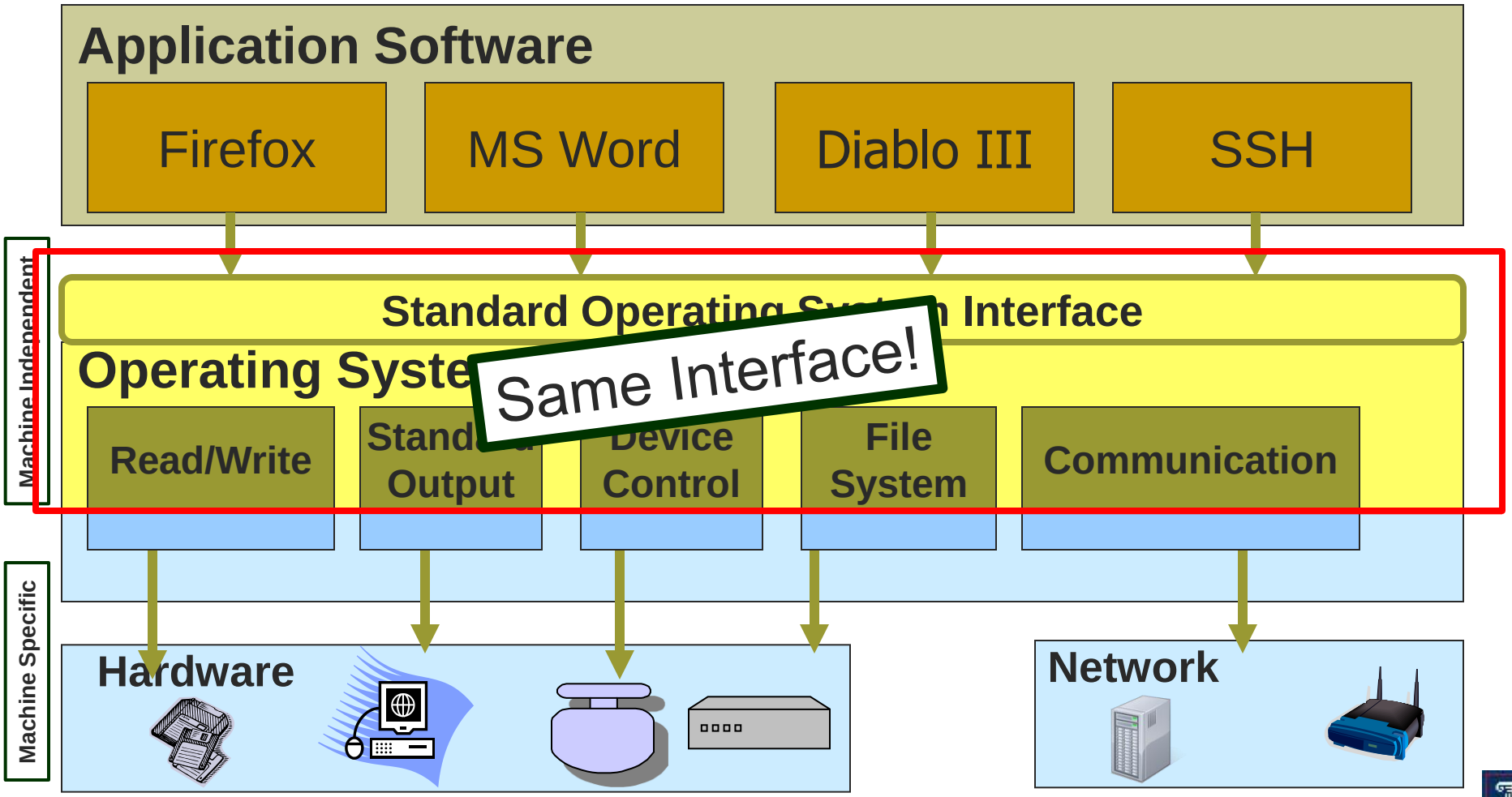
Machine Independent = Portable



OS Runs on Multiple Platforms



OS Runs on Multiple Platforms



POSIX
The UNIX Interface Standard