Think-piece

You are given a library, container.o and the following .h file signatures:

```
int put (int n, container* c); /* returns 1 and adds to c if n not in c,
  otherwise returns 0 */
int get (int n, container* c); /* returns 1 if n is in c, 0 otherwise */
int remove (int n, container* c); /* returns 1 if n was in c; after
  return n is not in c! */
container* newContainer(); /* returns a new container if memory avail */
```

You don't have source code, and the file isn't compiled with debugging information. Attached is a note: "We would like to use this (it's really fast) in our new system, but it needs to work well – a bug in this could be catastrophic. Can you give me a plan/approach for thorough testing? I don't want to share our test generation code with the company that wrote this, and they won't share source, but you can give them test cases. The programmer behind this at Container Code Design, LLC, is pretty busy, so we'd like to make sure to get good turnaround from debugging. Can I get a short white paper on this by this afternoon's 2:35 project meeting? I know it's short notice, and you're not really a test engineer, but we need something.

A bug! Missing quotation mark!

Think-piece

You are given a library, container.o and the following .h file signatures:

```
int put (int n, container* c); /* returns 1 and adds to c if n not in c,
  otherwise returns 0 */
int get (int n, container* c); /* returns 1 if n is in c, 0 otherwise */
int remove (int n, container* c); /* returns 1 if n was in c; after
  return n is not in c! */
container* newContainer(); /* returns a new container if memory avail */
```

You don't have source code, and the file isn't compiled with debugging information. Attached is a note: "We would like to use this (it's really fast) in our new system, but it needs to work well – a bug in this could be catastrophic. Can you give me a plan/approach for thorough testing? I don't want to share our test generation code with the company that wrote this, and they won't share source, but you can give them test cases. The programmer behind this at Container Code Design, LLC, is pretty busy, so we'd like to make sure to get good turnaround from debugging. Can I get a short white paper on this by this afternoon's 2:35 project meeting? I know it's short notice, and you're not really a test engineer, but we need something."

Think-piece

You are given a library, container.o and the following .h file signatures:

```
int put (int n, container* c); /* returns 1 and adds to c if n not in c,
  otherwise returns 0 */
int get (int n, container* c); /* returns 1 if n is in c, 0 otherwise */
int remove (int n, container* c); /* returns 1 if n was in c; after
  return n is not in c! */
container* newContainer(); /* returns a new container if memory avail */
```

You don't have source code, and the file isn't compiled with debugging information. Attached is a note: "We would like to use this (it's really fast) in our new system, but it needs to work well – a bug in this could be catastrophic. Can you give me a plan/approach for thorough testing? I don't want to share our test generation code with the company that wrote this, and they won't share source, but you can give them test cases. The programmer behind this at Container Code Design, LLC, is pretty busy, so we'd like to make sure to get good turnaround from debugging. Can I get a short white paper on this by this afternoon's 2:35 project meeting? I know it's short notice, and you're not really a test engineer, but we need something."

What Matters Here?

- Simplify the problem
- What we have:

```
int put (int n, container* c);
  returns 1 and adds to c if n not in c, otherwise returns 0

int get (int n, container* c);
  returns 1 if n is in c, 0 otherwise

int remove (int n, container* c);
  returns 1 if n was in c; after return n is not in c!

container* newContainer();
  returns a new container if memory avail
```

The API (Interface) We're Testing

- What is the goal of testing?

- We want to see if this code does what it says it does
- How can we do that?

```
int put (int n, container* c);
   returns 1 and adds to c if n not in c, otherwise
   returns 0

int get (int n, container* c);
   returns 1 if n is in c, 0 otherwise

int remove (int n, container* c);
   returns 1 if n was in c; after return n is not in
   c!

container* newContainer();
   returns a new container if memory avail
```

A Very Simple Test

- This is a typical manual unit test
  - Do something to the software that has a known result
  - *Assert* the result matches what you expect
- How much does this test?

```
c = newContainer();

r = put(0,c);
assert (r == 1);

r = put(0,c);
assert (r == 0);

r = get(0,c);
assert (r == 1);

r = remove(0,c);
assert (r == 1);

r = get(0,c);
assert (r == 0);
```

Manual Unit Tests

- How much does this test?
  - Probably not very much
  - That's ok, we can write more tests…
    - and more tests…
      - and still more tests…
  - How do we know we're done?

```
c = newContainer();

r = put(0,c);
assert (r == 1);

r = put(0,c);
assert (r == 0);

r = get(0,c);
assert (r == 1);

r = remove(0,c);
assert (r == 1);

r = get(0,c);
assert (r == 0);
```

Boredom Sets in Quickly

- Writing each sequence of operations we want to try is tedious
  - We're going to run out of patience before we try very many things
  - Each test takes a long time to write
  - Could we get the computer to do it for us?

Random Testing

- Here's an attempt:

```
c = newContainer();

for (int i = 0; i < NUM_TESTS; i++) {
 op = random(3);
 v = random(MAX_VALUE);
 if (op == 0)
   r1 = put(v,c);
 if (op == 1)
   r1 = get(v,c);
 if (op == 2)
   r1 = remove(v,c);
}
```

Random Testing

- What kind of bugs can this testing find?

```
c = newContainer();

for (int i = 0; i < NUM_TESTS; i++) {
 op = random(3);
 v = random(MAX_VALUE);
 if (op == 0)
   r1 = put(v,c);
 if (op == 1)
   r1 = get(v,c);
 if (op == 2)
   r1 = remove(v,c);
}
```

Differential Testing

- What does the container program act like?
  - A set
  - Do we have any other set implementations?
  - Could we write one that
    - we are pretty sure is correct
    - acts like our tested system is supposed to act?

Differential Testing

```
c = newContainer();
ref = newBinaryTree();
for (int i = 0; i < NUM_TESTS; i++) {
 op = random(3);
 v = random(MAX_VALUE);
 if (op == 0) {
   r1 = put(v,c);
   r2 = bt_put(v,ref);
 } else if (op == 1) {
   r1 = get(v,c);
   r2 = bt_get(v,ref);
 } else if (op == 2) {
   r1 = remove(v, c);
   r2 = bt_remove(v, ref);
 }
 assert (r1 == r2);
}
```

Software Maintenance

- This is not how software engineering works:
  - First, design happens,
  - Second, implementation happens,
  - Third, testing confirms implementation and design were successful
  - Fourth, the entire thing is frozen in carbonite forever and released to customers

Software Maintenance

- A second false idea is that the process of generating a working system is much more complicated than that, but
  - At some point “the working version” is complete, and after that most changes are (small) bug fixes.
  - This is the “software maintenance is like an oil change” theory: once in a while you have to make a small corrective action, but that’s it

Software Maintenance

- In fact, up to 80% of maintenance efforts are **not** bug fixes
  - Software systems that *are not* used simply die
  - Software systems that *are* used evolve to match their user environment
    - Adding features
  - Adding features adds complexity
  - Adding complexity requires occasional *re-factoring* unless you want to end up with code that requires a "software archaeologist" to understand

Source Control

- How do we manage all these changes?
  - Real-world projects are not a set of three .java files, or two .c files and two .h files
  - Real-world systems are complicated trees of source files, support files, documentation, test cases, and configuration files
  - Multiple developers work on the tree and make changes to it
    - May want to go back to an old version
    - May want to work on a file when you don't have access to a shared network location
    - Dropbox/copying around a zipped version is clumsy and prone to disaster

Source Control Systems

- Use **source control systems** (also known as revision control or version control)
- Common examples:
  - rcs
  - cvs
  - svn (subversion)
  - git
- We're going to use a small subset of svn, which I will introduce on the next few slides
  - svn is widely available and has a simple model

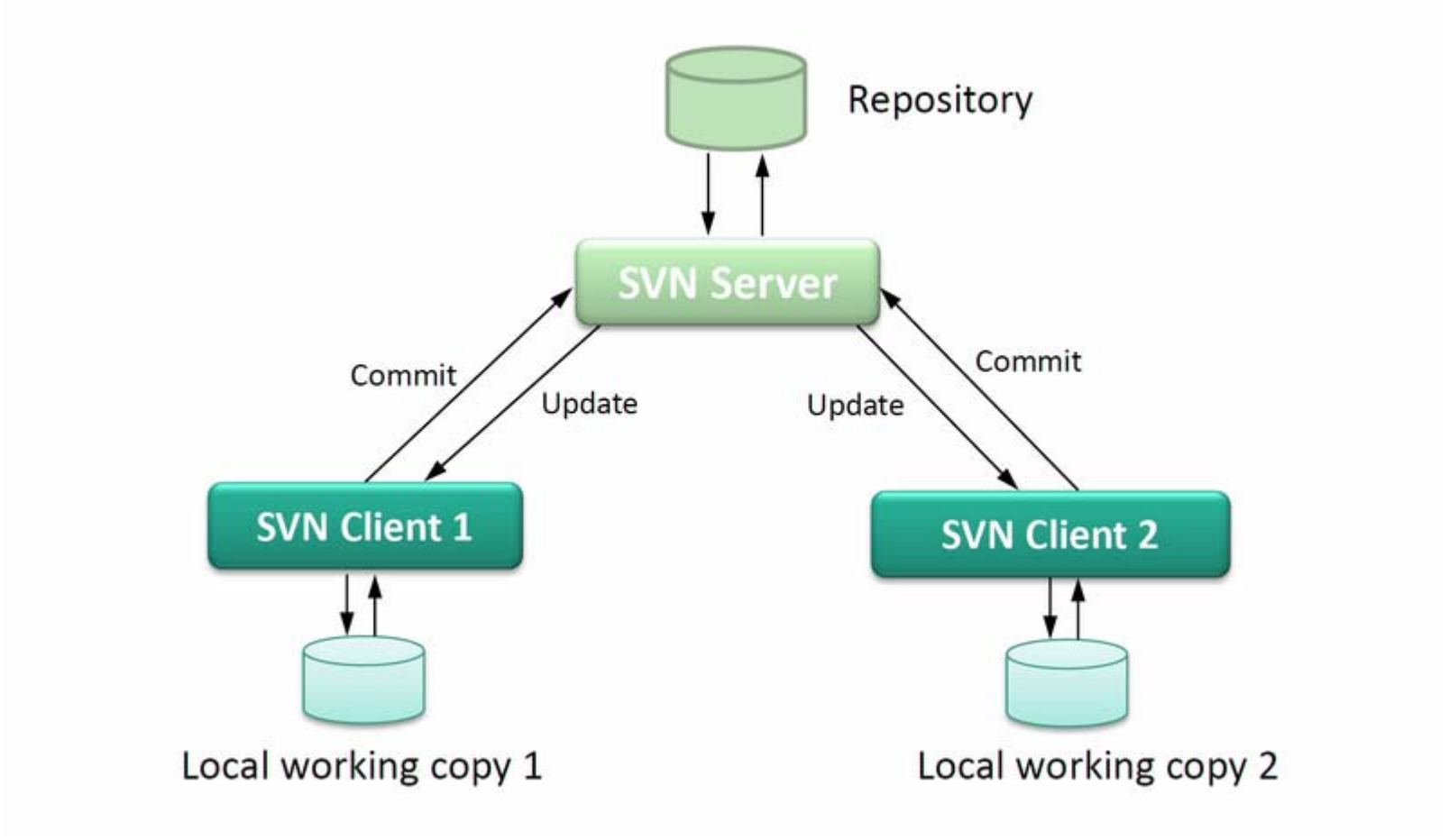
Intro to svn

- All source control systems:
  - Track changes made to a software system
  - Allow merging of changes to a part of a system
  - Allow the development of multiple versions of a system

- We’re going to look at just three operations:
  - Checking out a project
  - Checking changes into a project
  - Updating a project to get the latest changes made by others

  - Also look at the web interface to svn provided by code.google.com

Intro to svn


Repository
SVN Server
Commit
Update
Update
Commit
SVN Client 1
SVN Client 2
Local working copy 1
Local working copy 2

Getting Started with svn

- Log in to code.google.com (use your ONID account)

- Go to a command prompt (cygwin or unix/linux)
  - Navigate till your current directory is a location you want to store your class code repository
  - **svn checkout *https*://cs362summer14.googlecode.com/svn/trunk/ cs362summer14 --username XXXXX@onid.oregonstate.edu**
  - A new directory will be created below your current directory location, containing the dominion source code
  - This is the contents of the repository

Creating Your Own Project Space

- Navigate into the cs362class directory projects section
  - `cd cs362class/projects`
- Create your own directory for your code and tests:
  - `mkdir <onid username>`
- Is your code in the repository where everyone can see it now?
  - Navigate in your web browser to the projects directory

Making Your Work Visible to Others

- How do you add your directory to the repository?
  - `svn add <onid username>`
- Is your code in the repository *now?*
- Why not??
  - Adding a directory or file doesn't actually put it in the repository, it just tells svn you will eventually *check it i*

- One more step:
  - `svn commit <onid username> -m "Initial checkin"`

Failed Permissions?

- If this didn't work, you probably don't have permission to add things to the class repository
  - Need to contact me.

Seeing the Work of Other People

- Other people may change their files, add files, etc.
- How do you get the latest version of everything?
- Navigate to the top of the repository (or where ever you want to update everything below) and try:
  - `svn update`

SVN REMINDERS

- You can look on google code to see if your code is actually checked in, and what version it is
- You need to add every file and directory you want other people to be able to look at
  - If you add a directory svn will automatically try to add all the files in that directory
- Just adding things doesn't actually put it in the repository
- You have to *check in* your changes