# Classes and Objects

## Design

How to translate problem into classes and methods?

Initially seems harder than finding functions

"With great power comes great responsibility"

Introduce a simple design process

Always ask: what is this class responsible for?

A one-phrase summary of what the class takes care

of that nothing else in the program takes care of

Discover/create these using *CRC cards*

. Class

. Responsibility

. Collaboration

| InterpolatedSignal | |
|---|---|
| - manage sampled values<br><br>- find interpolation point | - StepSignal<br><br>- LinearSignal |

Example: audiology experiments

*Clinician* applies *test* to *subject* to get *result*

*Test* specifies what to do and what scores to collect
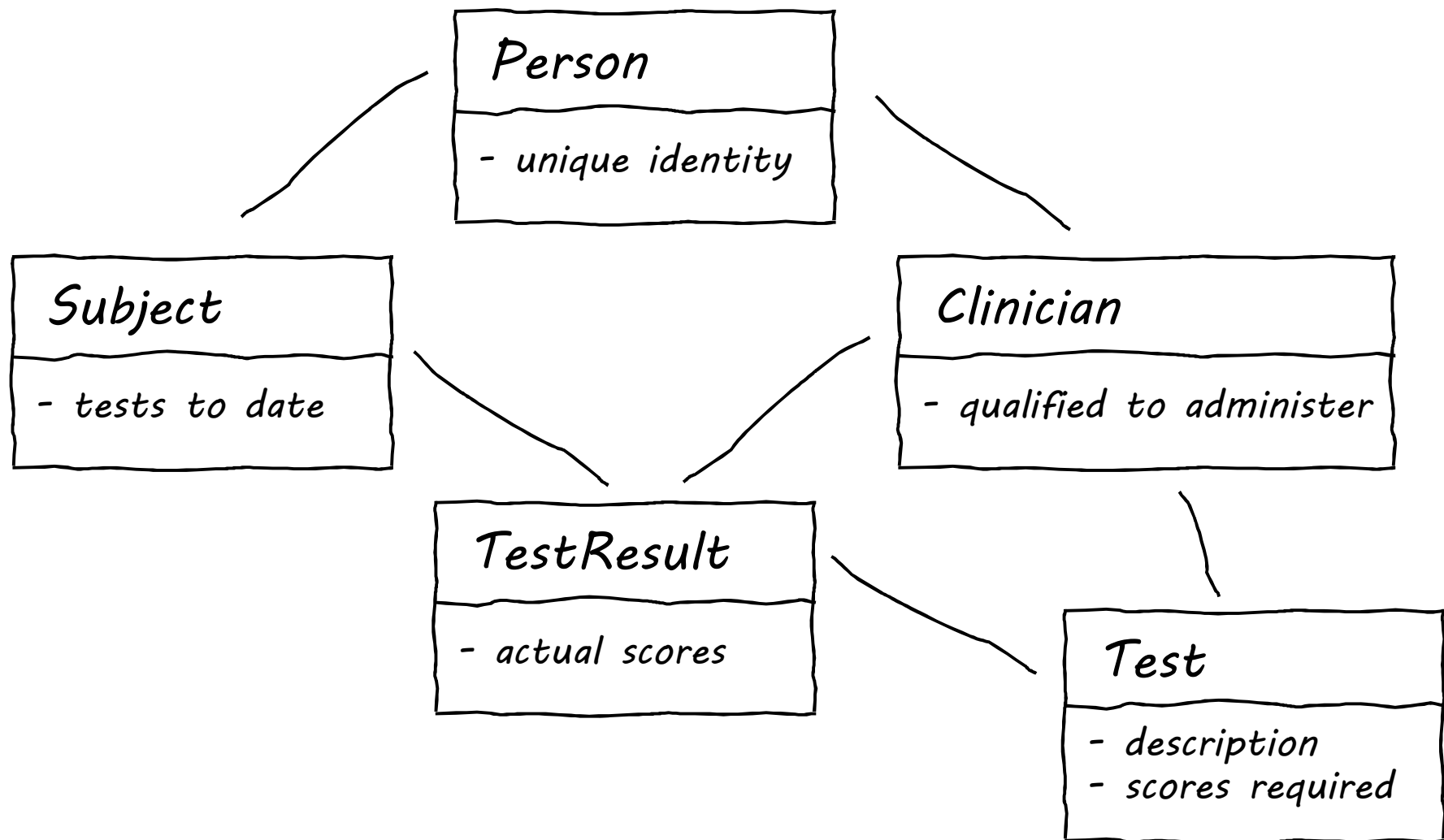
*Clinicians* and *subjects* are people

*Clinician* is only qualified to administer certain *tests*
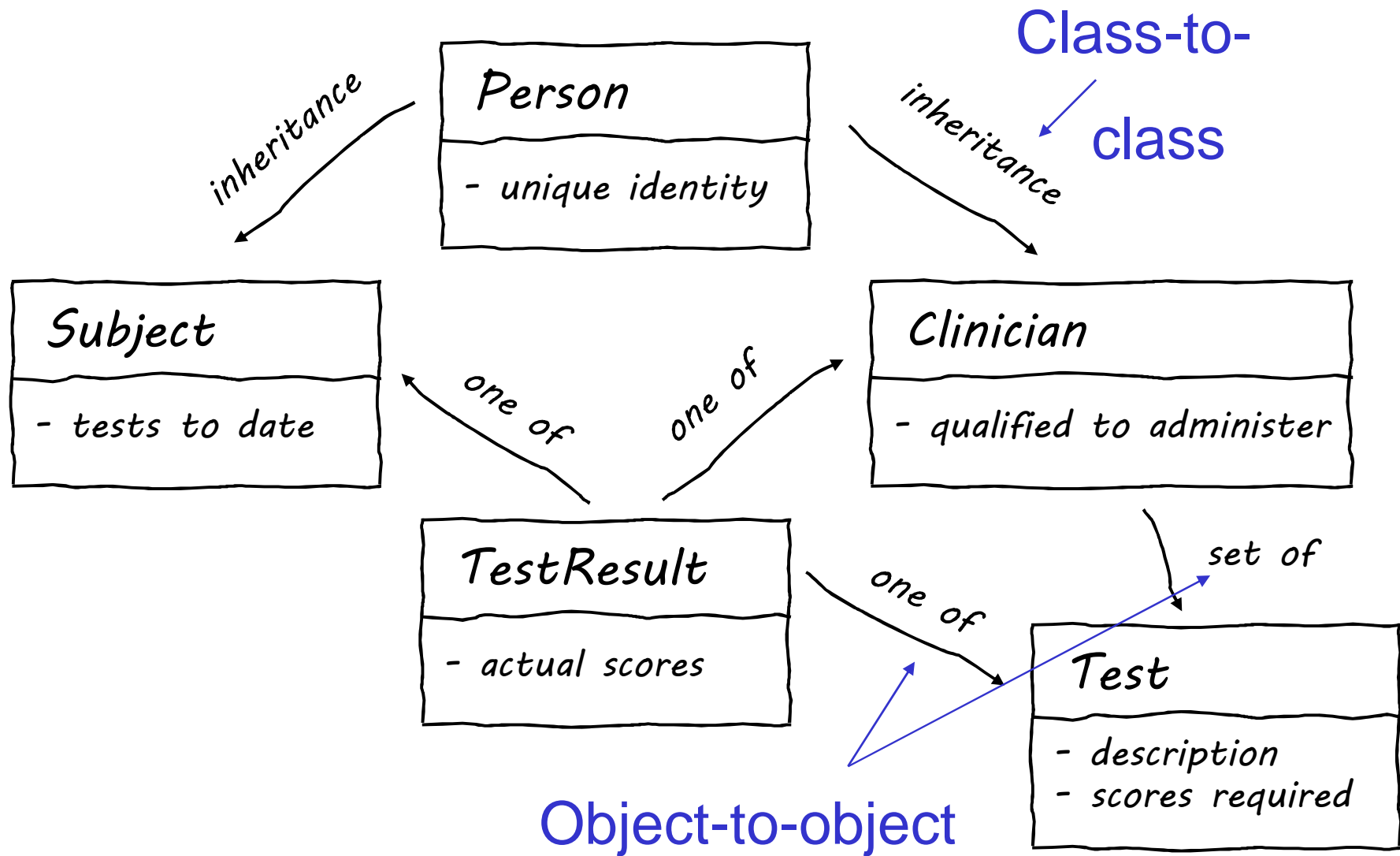
First attempt at design: look for nouns and verbs

Nouns are classes and objects

Verbs are collaborations and methods

# Show collaborations with lines to save space

**Person**

- unique identity

**Subject**

- tests to date

**Clinician**

- qualified to administer

**TestResult**

- actual scores

**Test**

- description
- scores required

# What kind of relationship is each?

**Class-to-class**

**Person**
- unique identity

*inheritance*

*inheritance*

**Subject**
- tests to date

**Clinician**
- qualified to administer

*one of*

*one of*

**TestResult**
- actual scores

*one of*

*set of*

**Test**
- description
- scores required

**Object-to-object**

Mistake #1: formalization

. The cheaper it looks, the more honest people are

. Use 3×5 index cards or sticky notes

Mistake #2: members and methods instead of responsibilities and collaborations

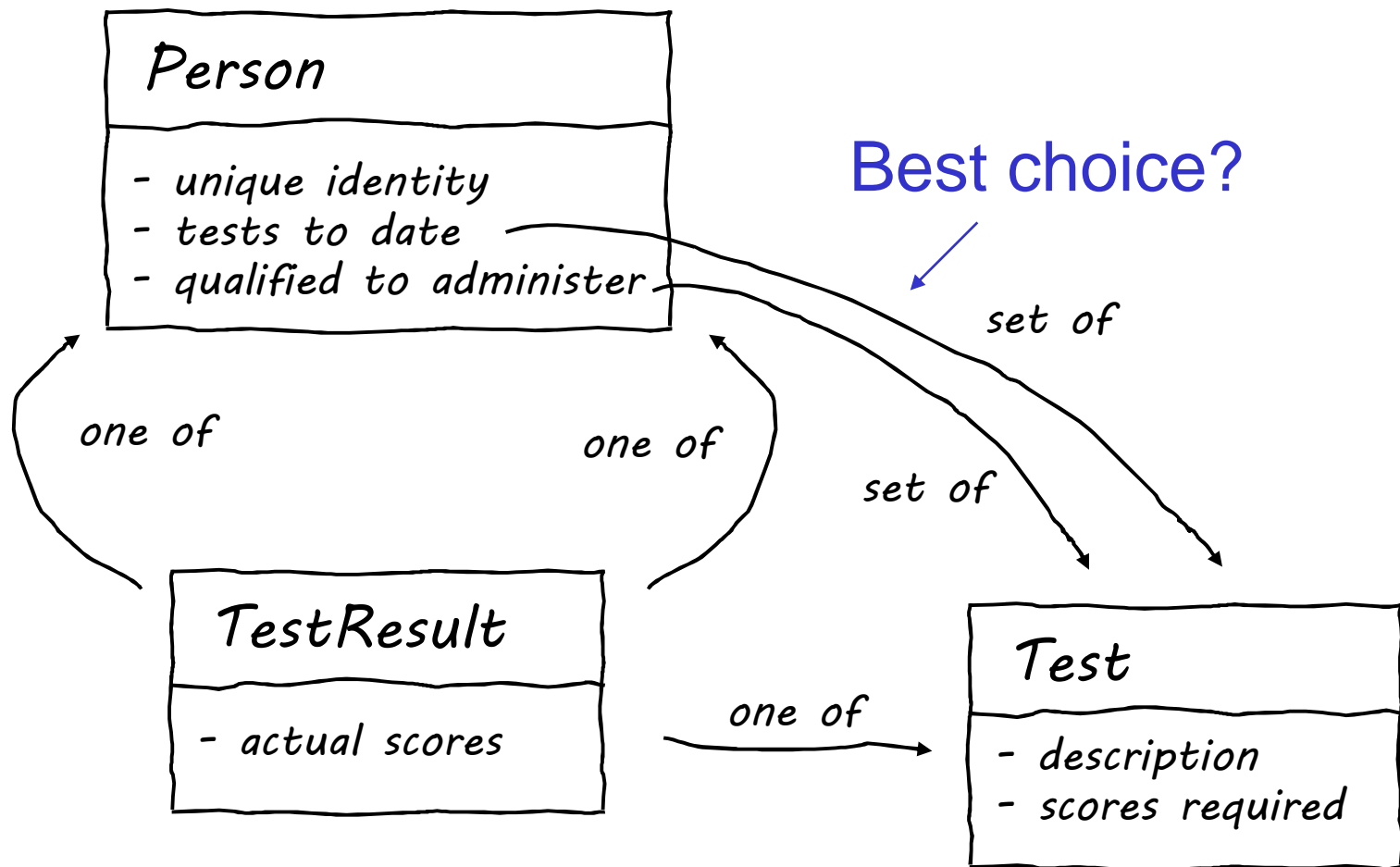A doctor is not "someone in a white coat with a stethoscope"

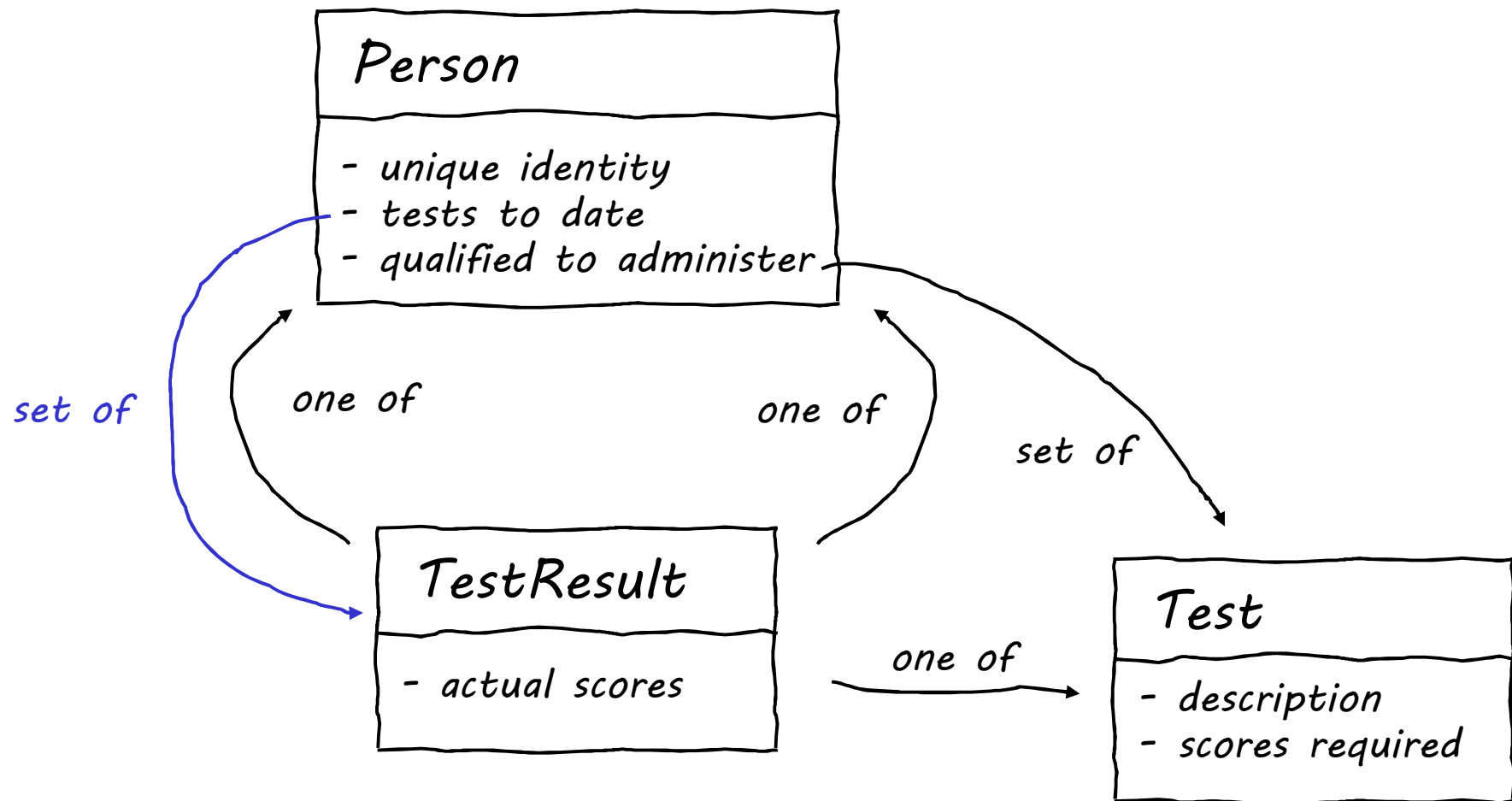Can a clinician also be a subject?   Yes

Poor fit for this design:

.   One object for person-as-subject

.   A different object for person-as-clinician

Put both sets of information into `Person` class

# New set of classes

**Person**

- unique identity
- tests to date
- qualified to administer

**Best choice?**

set of

one of

one of

set of

**TestResult**

- actual scores

one of
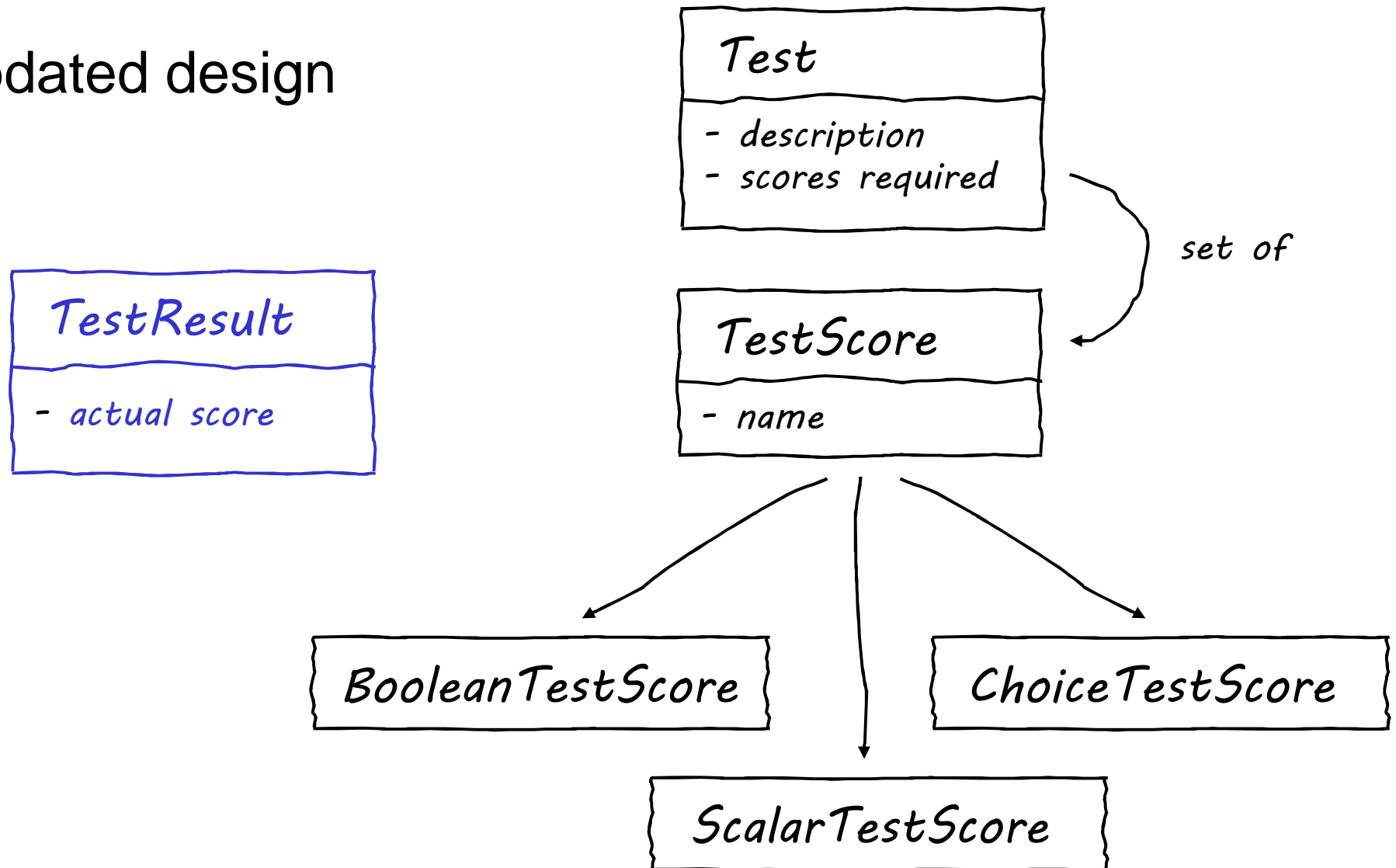
**Test**

- description
- scores required

# Improvement

Take a closer look at test scores

. True/False

. Scalar (e.g., frequency or volume threshold)

. Choice (e.g., favorite from list of N items)

Use inheritance

. Generic properties of test scores in parent class

. One subclass for each kind of test score

# Updated design

**Test**

- description
- scores required

*set of*

**TestResult**

- actual score

**TestScore**

- name

**BooleanTestScore**

**ScalarTestScore**

**ChoiceTestScore**

Can't finish until we ~~know~~ how TestResult will be

used                               <span style="color:#990000">decide</span>

Do all test results share generic responsibilities?

.   load/display/edit/save

.   convert to (weighted?) numeric value

*Every* complete program *will have made* decisions

"A week of hard work can sometimes save you

an hour of thought"

Think this through *before* you start coding

Methods and members follow naturally once responsibilities are assigned

Classes should never be designed in isolation

.   Collaborate with each other

.   Distill experience

created by

# Greg Wilson

December 2010