Confessions of a Test Driven Developer-holic

by Carlus R. Henry Sr.
carlushenry@gmail.com

THANKS TO ALL OUR SPONSORS!

Who am I?

- Carlus R. Henry Sr.
- 5’4”

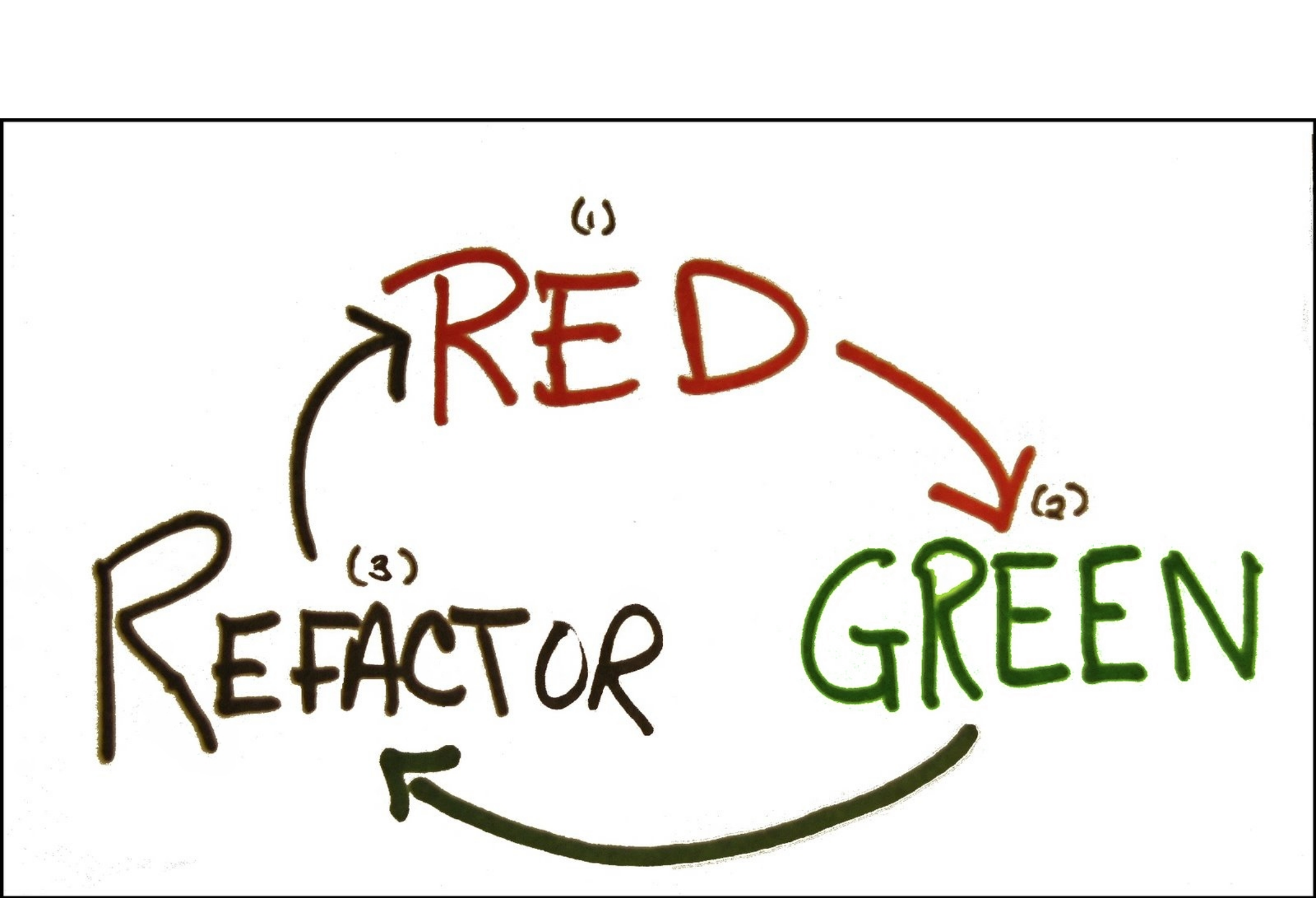
Who am I?

- Carlus R. Henry Sr.
- ~~5’4”~~ okay, okay….5’2”
- Husband
- Father of 6
- Consultant 12 years
- Manager at KPMG (3 weeks)

Overview

- My Test Driven Development Journey
- “Moment of Clarity”
- Better Way

What is Test Driven Development?


(1)
RED
(2)
GREEN
(3)
REFACTOR

Life without TDD


OOPS... MY BAD!
memegenerator.net

Life with TDD

Moment of Clarity

Moment of Clarity

They are rare. Nevertheless, everybody has them. Those unanticipated seconds in time when the whirlwind of life ceases and a virgin oasis of awareness suddenly opens the mind to a thought or a vision that resonates beyond that moment, even when the moment goes away…

– Christopher Kennedy Lawford

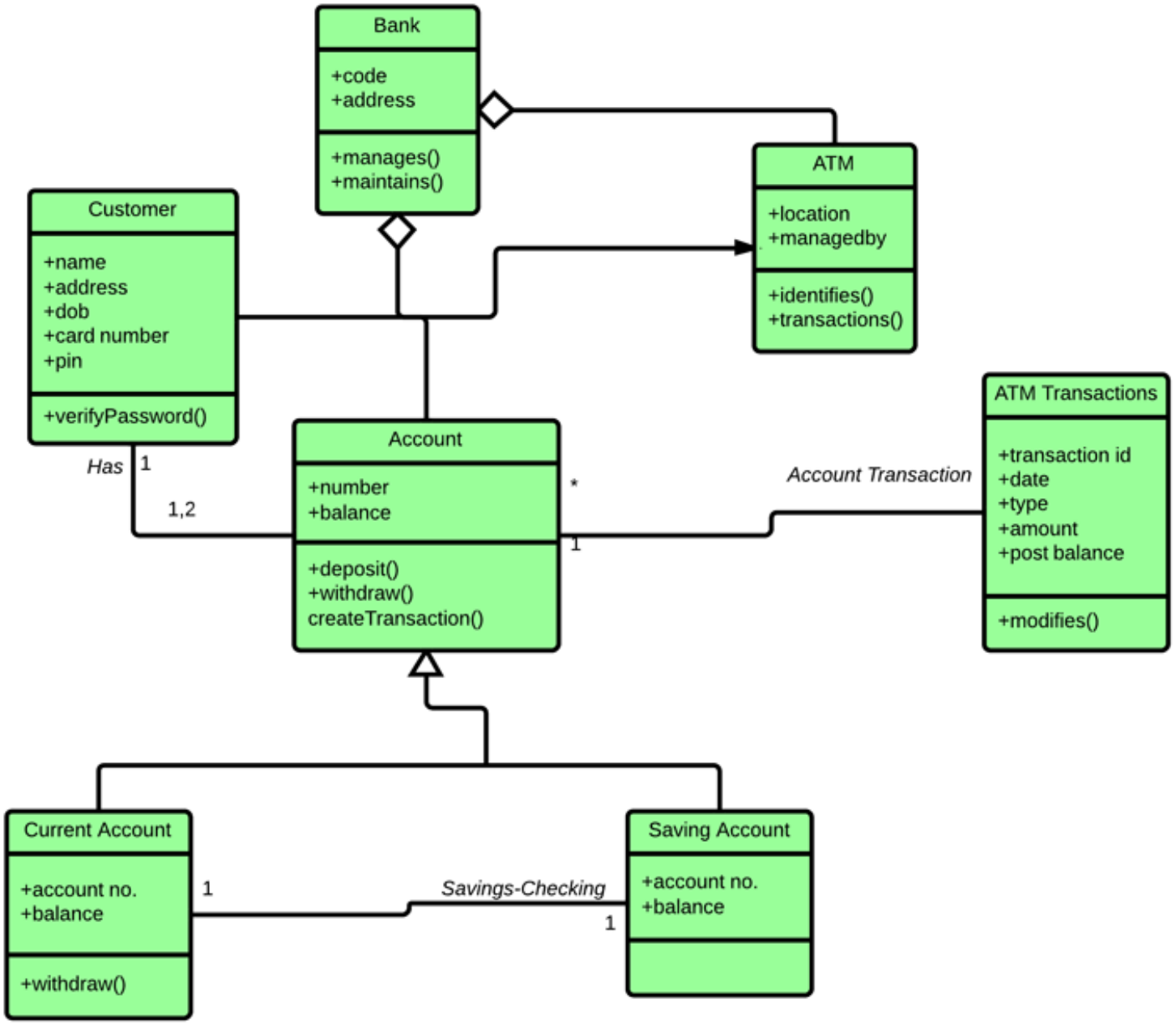
“Unit Tests Suck”

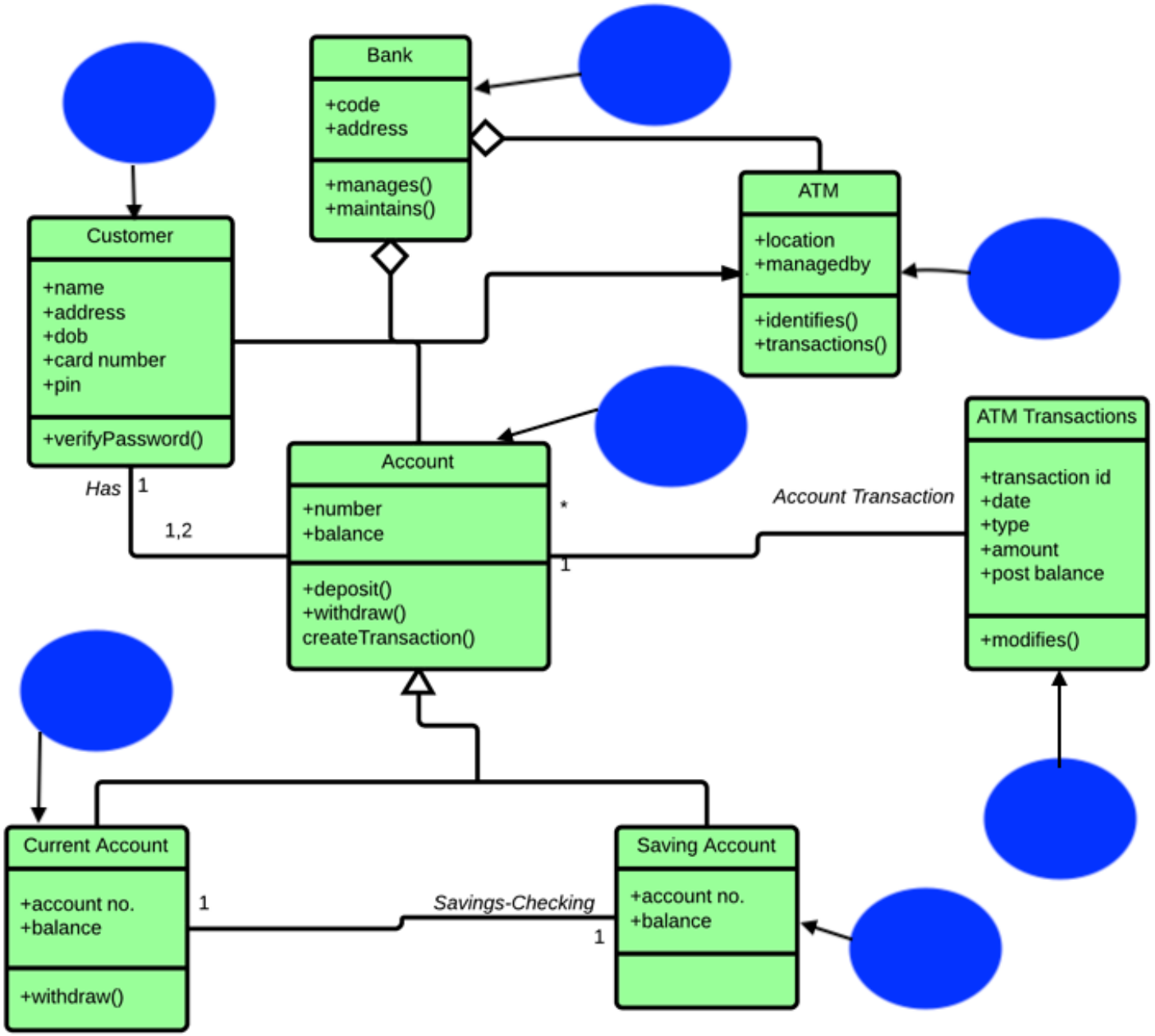

HOW DARE YOU?!?!?
memegenerator.net

Pitfalls of TDD / Unit Tests

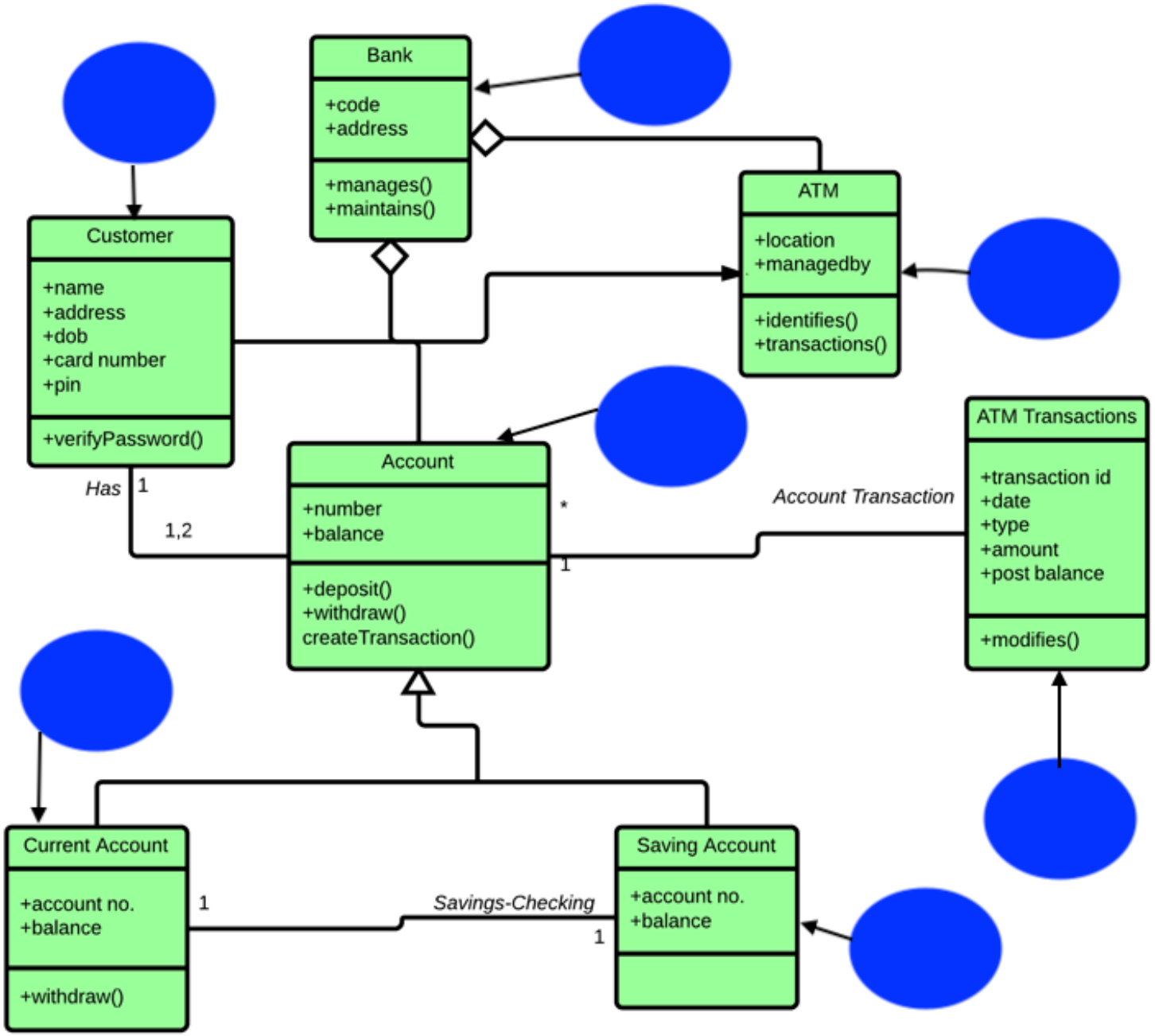
- Don’t Prove your System Works
- Discourage refactoring
- Mocking is a Horrible Idea

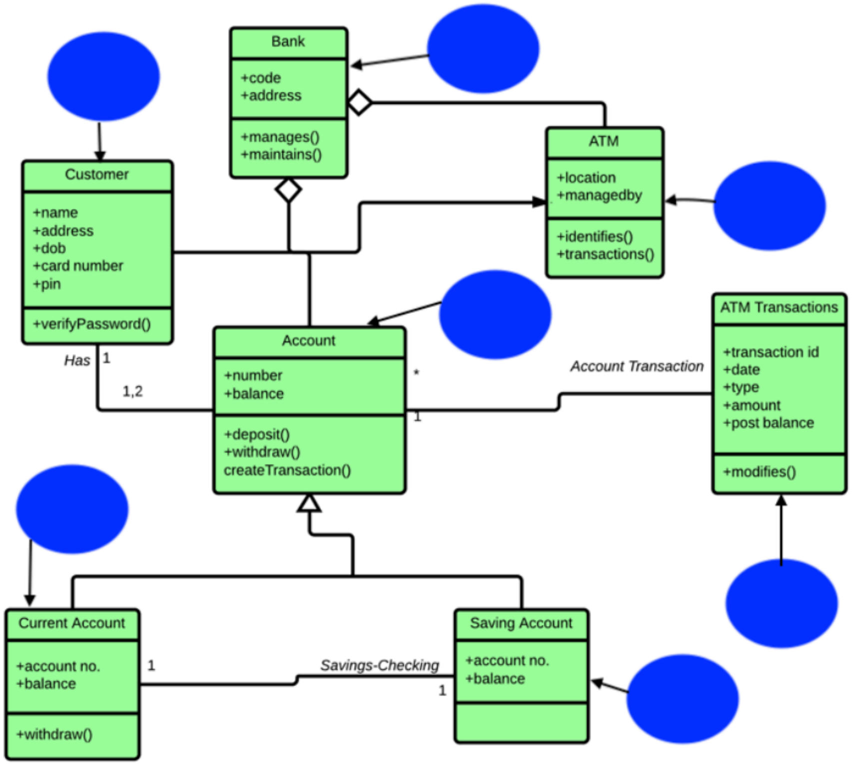
Don’t Prove your System Works

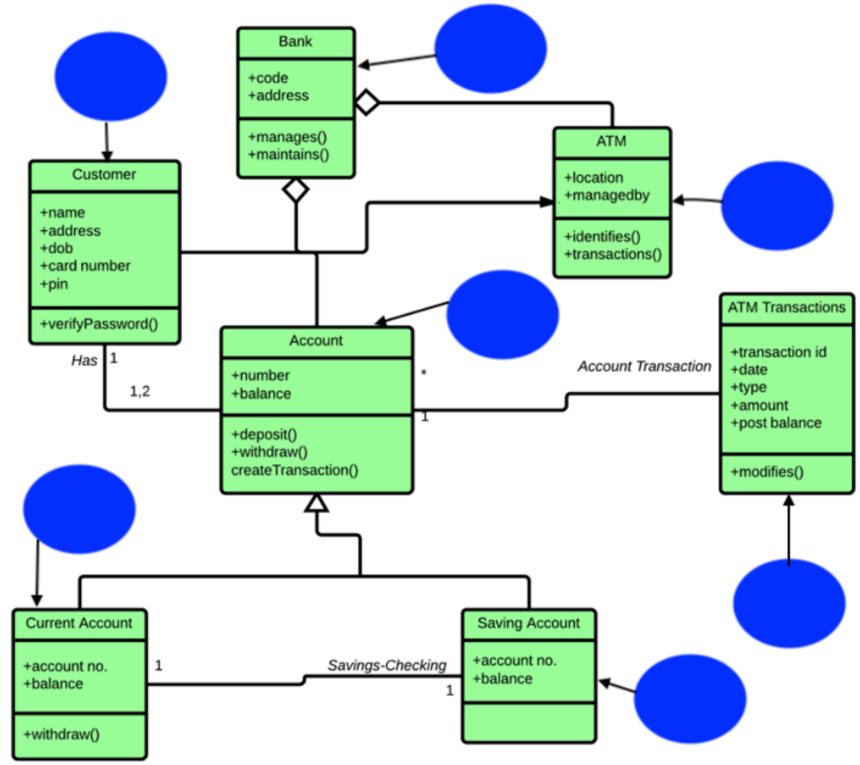

Bank
+code
+address
+manages()
+maintains()
ATM
+location
+managedby
+identifies()
+transactions()
Customer
+name
+address
+dob
+card number
+pin
+verifyPassword()
Has
1
1,2
Account
+number
+balance
+deposit()
+withdraw()
createTransaction()
*
1
Account Transaction
ATM Transactions
+transaction id
+date
+type
+amount
+post balance
+modifies()
Current Account
+account no.
+balance
+withdraw()
1
Savings-Checking
1
Saving Account
+account no.
+balance

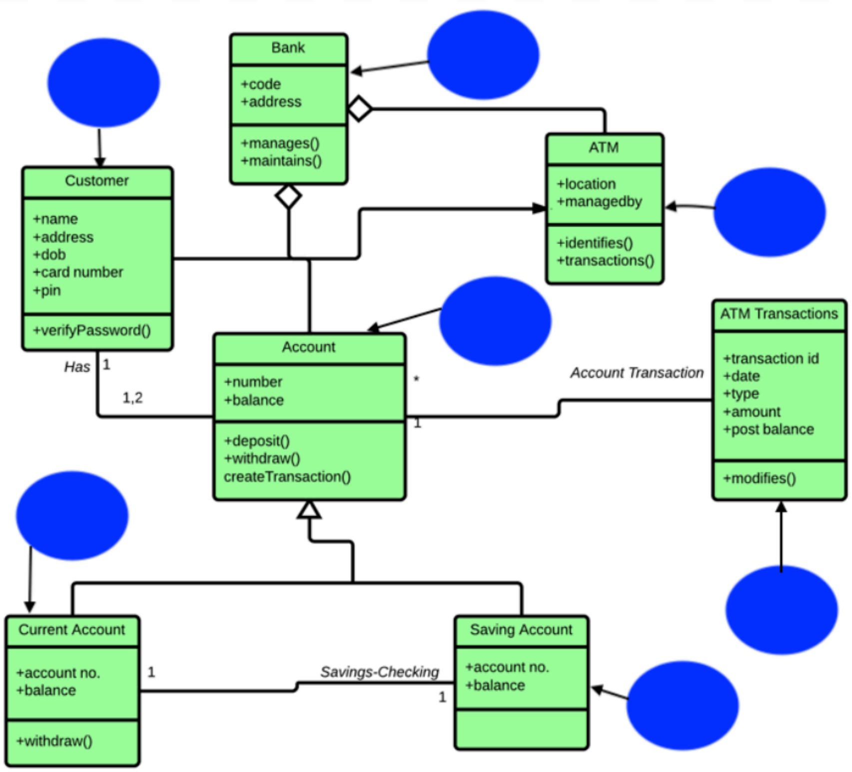

Bank
+code
+address
+manages()
+maintains()
Customer
+name
+address
+dob
+card number
+pin
+verifyPassword()
ATM
+location
+managedby
+identifies()
+transactions()
Account
+number
+balance
+deposit()
+withdraw()
createTransaction()
ATM Transactions
+transaction id
+date
+type
+amount
+post balance
+modifies()
Has
1
1,2
*
1
Account Transaction
Current Account
+account no.
+balance
+withdraw()
1
Savings-Checking
1
Saving Account
+account no.
+balance

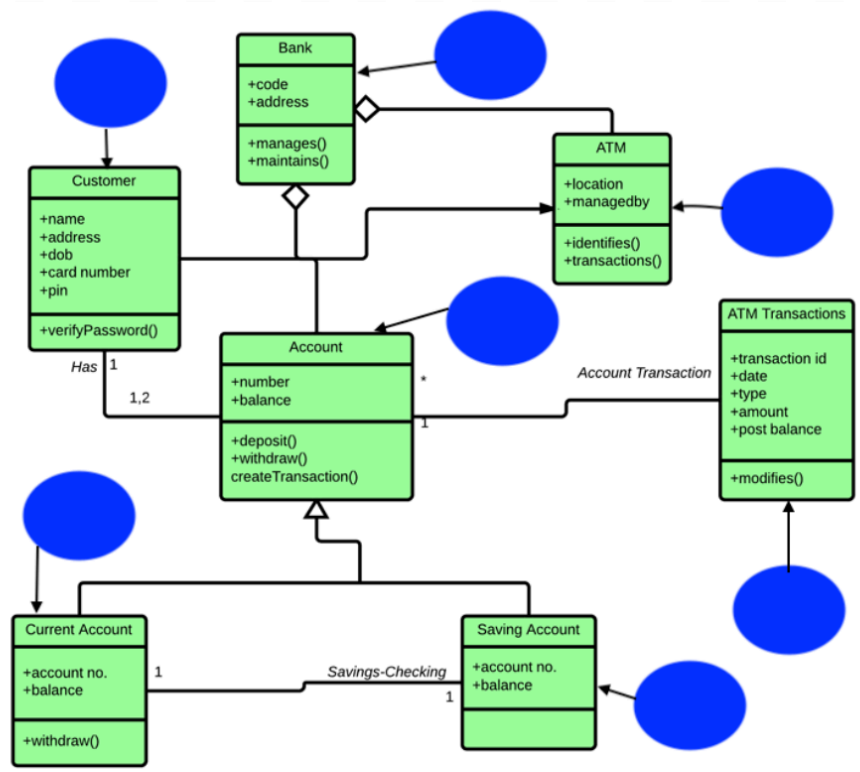
Discourage Refactoring


Bank
+code
+address
+manages()
+maintains()
Customer
+name
+address
+dob
+card number
+pin
+verifyPassword()
ATM
+location
+managedby
+identifies()
+transactions()
Account
+number
+balance
+deposit()
+withdraw()
createTransaction()
ATM Transactions
+transaction id
+date
+type
+amount
+post balance
+modifies()
Has
1
1,2
*
1
Account Transaction
Current Account
+account no.
+balance
+withdraw()
1
Savings-Checking
1
Saving Account
+account no.
+balance

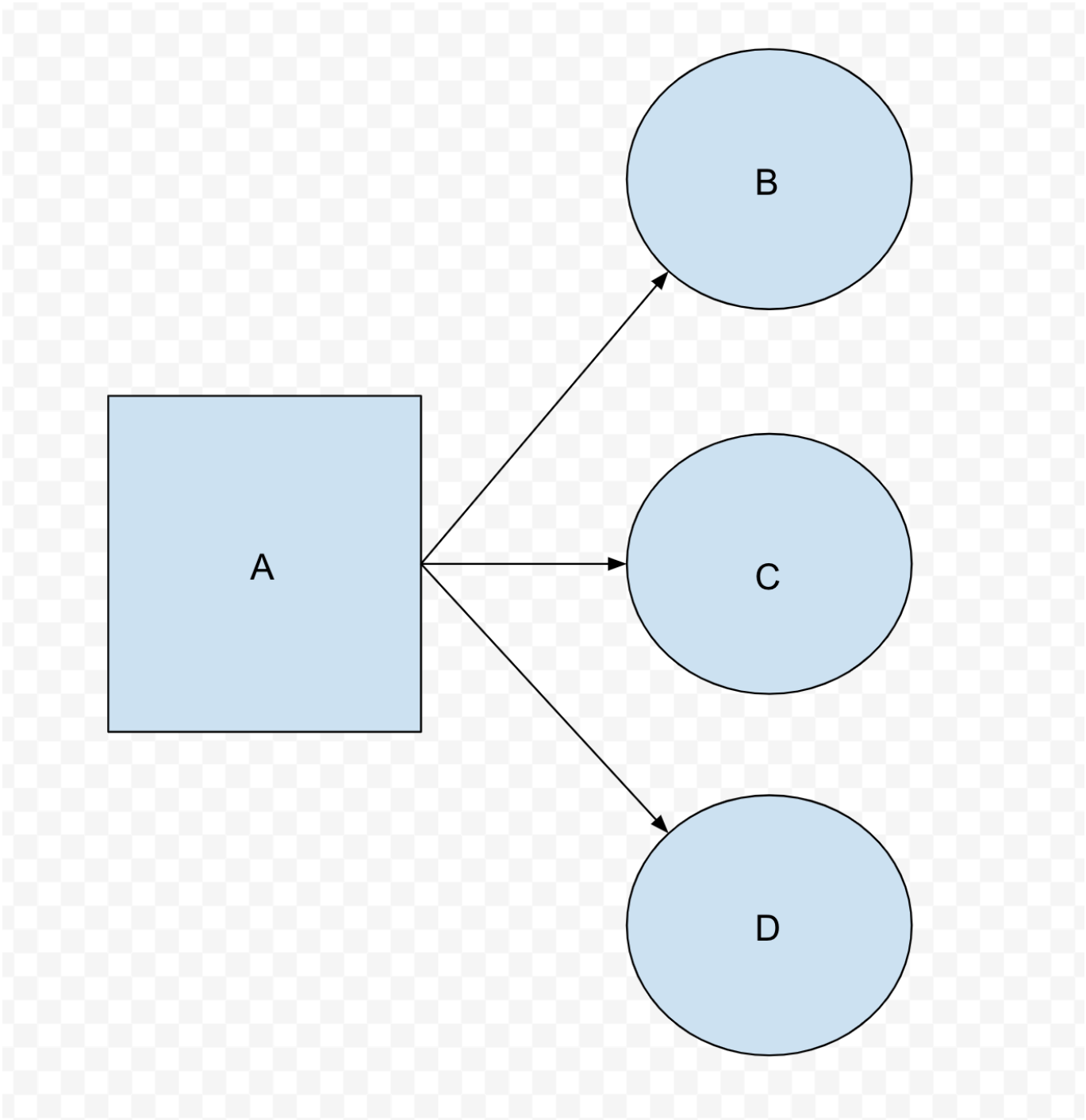

Bank
+code
+address
+manages()
+maintains()
Customer
+name
+address
+dob
+card number
+pin
+verifyPassword()
ATM
+location
+managedby
+identifies()
+transactions()
Account
+number
+balance
+deposit()
+withdraw()
createTransaction()
ATM Transactions
+transaction id
+date
+type
+amount
+post balance
+modifies()
Has
1
1,2
*
1
Account Transaction
Current Account
+account no.
+balance
+withdraw()
Saving Account
+account no.
+balance
1
Savings-Checking
1


Bank
+code
+address
+manages()
+maintains()
Customer
+name
+address
+dob
+card number
+pin
+verifyPassword()
ATM
+location
+managedby
+identifies()
+transactions()
Account
+number
+balance
+deposit()
+withdraw()
createTransaction()
ATM Transactions
+transaction id
+date
+type
+amount
+post balance
+modifies()
Has
1
1,2
*
1
Account Transaction
Current Account
+account no.
+balance
+withdraw()
Saving Account
+account no.
+balance
1
Savings-Checking
1


Bank
+code
+address
+manages()
+maintains()
Customer
+name
+address
+dob
+card number
+pin
+verifyPassword()
ATM
+location
+managedby
+identifies()
+transactions()
Account
+number
+balance
+deposit()
+withdraw()
createTransaction()
ATM Transactions
+transaction id
+date
+type
+amount
+post balance
+modifies()
Has
1
1,2
*
1
Account Transaction
Current Account
+account no.
+balance
+withdraw()
Saving Account
+account no.
+balance
1
Savings-Checking
1


Bank
+code
+address
+manages()
+maintains()
Customer
+name
+address
+dob
+card number
+pin
+verifyPassword()
ATM
+location
+managedby
+identifies()
+transactions()
Account
+number
+balance
+deposit()
+withdraw()
createTransaction()
ATM Transactions
+transaction id
+date
+type
+amount
+post balance
+modifies()
Has
1
1,2
*
1
Account Transaction
Current Account
+account no.
+balance
+withdraw()
Saving Account
+account no.
+balance
1
Savings-Checking
1

Mocking is Horrible


B
A
C
D


THE BETTER WAY
JUST AHEAD

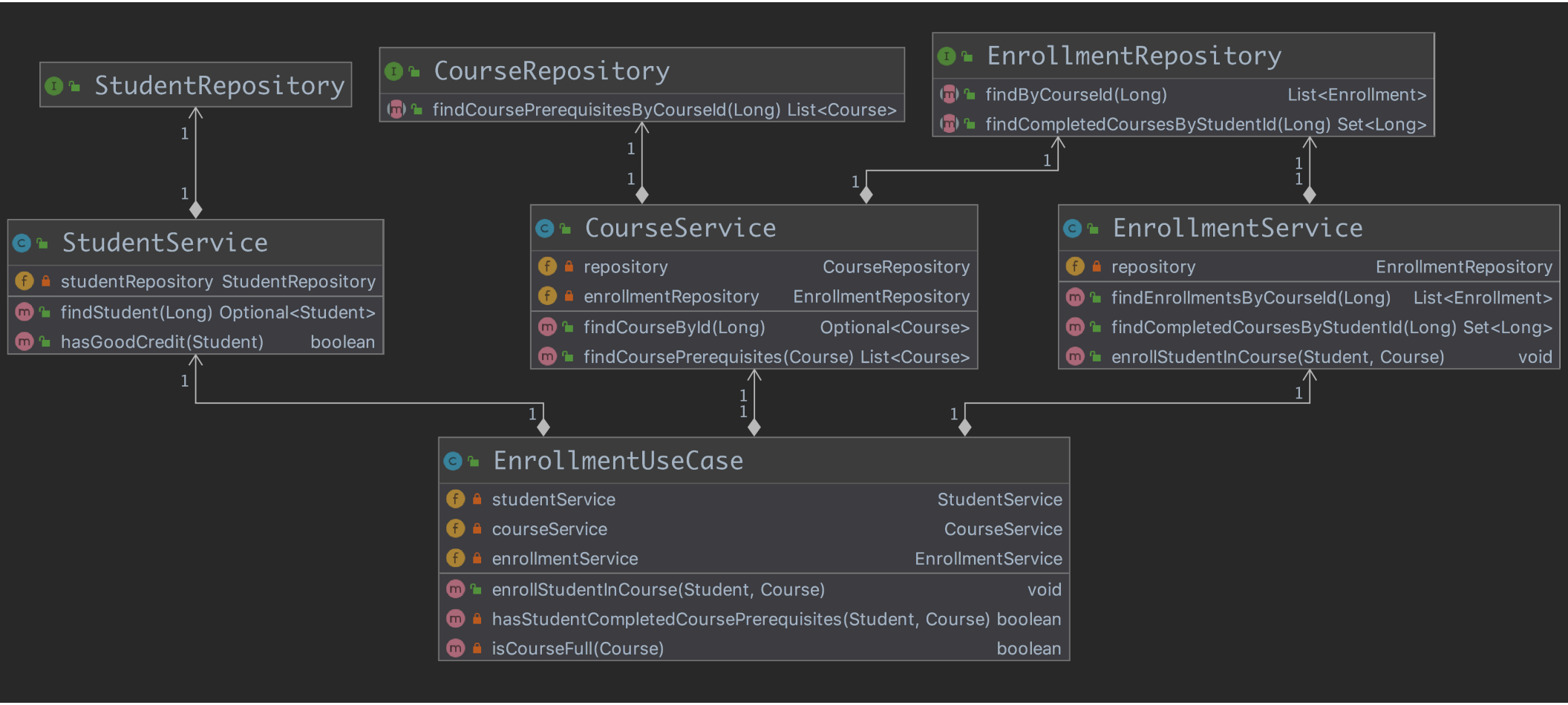
The Better Way

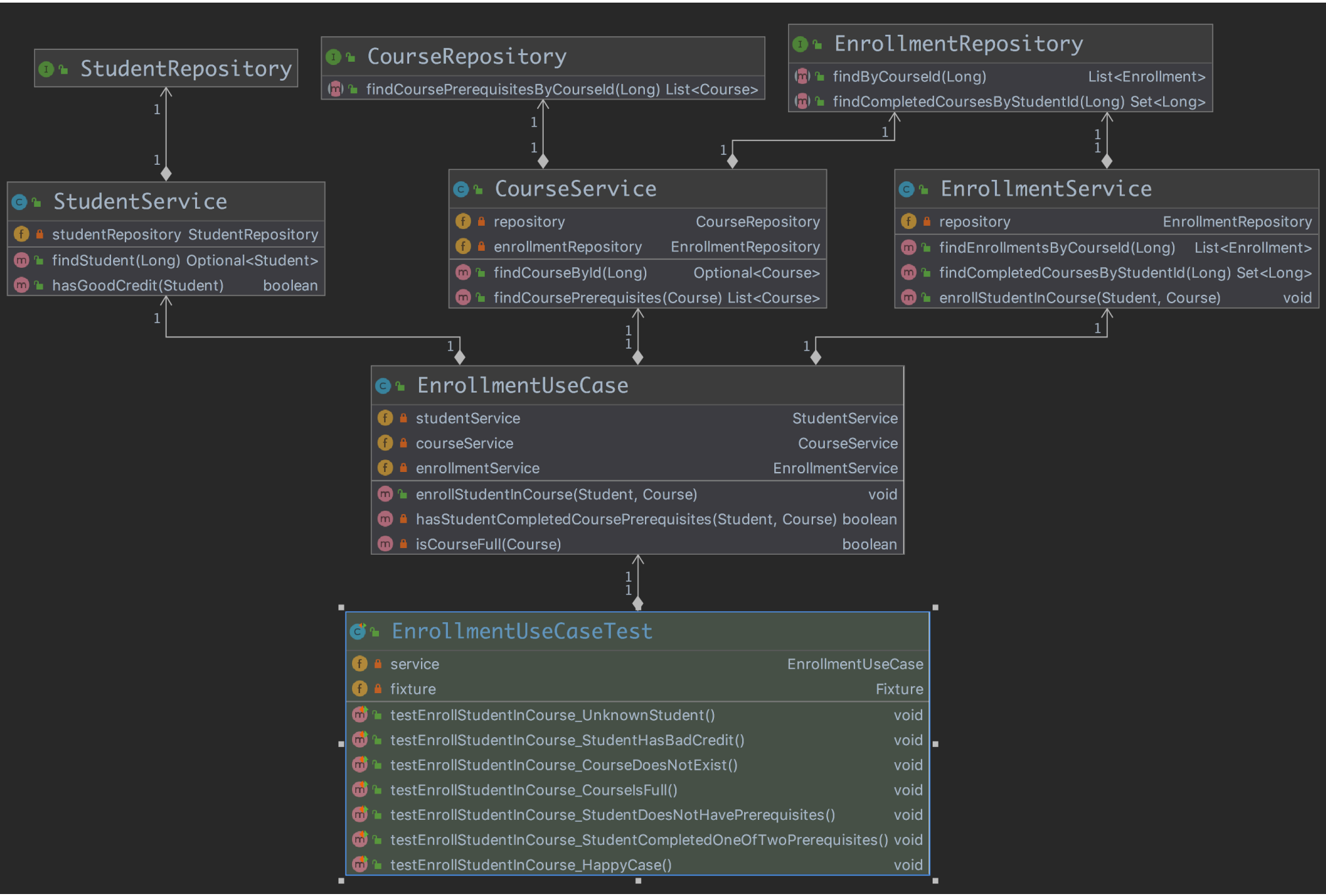
- Automated Tests that Test the Entire System
- Does not get in the way of Refactoring
- Tests should not be complex - easily understood

Better Way Example

https://github.com/chenry/studentenrollment

```sql
create table student (
    ID BIGINT PRIMARY KEY,
    FIRST_NAME VARCHAR2,
    LAST_NAME VARCHAR2,
    CREDIT_RATING INT
);

create table course (
    ID BIGINT PRIMARY KEY,
    COURSE_NAME VARCHAR2,
    DESCRIPTION VARCHAR2,
    CLASS_SIZE_LIMIT INT
);

create table course_prerequisite (
    ID BIGINT PRIMARY KEY,
    COURSE_ID BIGINT,
    PREREQ_COURSE_ID BIGINT,
    FOREIGN KEY (COURSE_ID) REFERENCES COURSE(ID),
    FOREIGN KEY (PREREQ_COURSE_ID) REFERENCES COURSE(ID)
);

create sequence enrollment_seq start with 1 increment by 1;

CREATE TABLE ENROLLMENT (
    ID BIGINT PRIMARY KEY,
    STUDENT_ID BIGINT,
    COURSE_ID BIGINT,
    IS_COMPLETED BOOLEAN,
    FOREIGN KEY (STUDENT_ID) REFERENCES STUDENT(ID),
    FOREIGN KEY (COURSE_ID) REFERENCES COURSE(ID)
);
```


StudentRepository
CourseRepository
findCoursePrerequisitesByCourseId(Long) List<Course>
EnrollmentRepository
findByCourseId(Long) List<Enrollment>
findCompletedCoursesByStudentId(Long) Set<Long>
StudentService
studentRepository StudentRepository
findStudent(Long) Optional<Student>
hasGoodCredit(Student) boolean
CourseService
repository CourseRepository
enrollmentRepository EnrollmentRepository
findCourseById(Long) Optional<Course>
findCoursePrerequisites(Course) List<Course>
EnrollmentService
repository EnrollmentRepository
findEnrollmentsByCourseId(Long) List<Enrollment>
findCompletedCoursesByStudentId(Long) Set<Long>
enrollStudentInCourse(Student, Course) void
EnrollmentUseCase
studentService StudentService
courseService CourseService
enrollmentService EnrollmentService
enrollStudentInCourse(Student, Course) void
hasStudentCompletedCoursePrerequisites(Student, Course) boolean
isCourseFull(Course) boolean


StudentRepository
CourseRepository
findCoursePrerequisitesByCourseId(Long) List<Course>
EnrollmentRepository
findByCourseId(Long) List<Enrollment>
findCompletedCoursesByStudentId(Long) Set<Long>
StudentService
studentRepository StudentRepository
findStudent(Long) Optional<Student>
hasGoodCredit(Student) boolean
CourseService
repository CourseRepository
enrollmentRepository EnrollmentRepository
findCourseById(Long) Optional<Course>
findCoursePrerequisites(Course) List<Course>
EnrollmentService
repository EnrollmentRepository
findEnrollmentsByCourseId(Long) List<Enrollment>
findCompletedCoursesByStudentId(Long) Set<Long>
enrollStudentInCourse(Student, Course) void
EnrollmentUseCase
studentService StudentService
courseService CourseService
enrollmentService EnrollmentService
enrollStudentInCourse(Student, Course) void
hasStudentCompletedCoursePrerequisites(Student, Course) boolean
isCourseFull(Course) boolean
EnrollmentUseCaseTest
service EnrollmentUseCase
fixture Fixture
testEnrollStudentInCourse_UnknownStudent() void
testEnrollStudentInCourse_StudentHasBadCredit() void
testEnrollStudentInCourse_CourseDoesNotExist() void
testEnrollStudentInCourse_CourseIsFull() void
testEnrollStudentInCourse_StudentDoesNotHavePrerequisites() void
testEnrollStudentInCourse_StudentCompletedOneOfTwoPrerequisites() void
testEnrollStudentInCourse_HappyCase() void

Show Code