Software Regression Testing

Speaker: Jerry Gao Ph.D.

Computer Engineering Department
San Jose State University

email: jerry.gao@sjsu.edu
URL: http://www.engr.sjsu.edu/gaojerry

Presentation Outline

- *What is Software Regression Testing?*
- *Basic Software Regression Problems*
- *Software Regression Testing Process*
- *Regression Strategies for Traditional Software*
- *Basic Solutions to Software Regression Problems*
- *Regression Strategies for Object-Oriented Software*

What is Software Regression Testing?

What is Software Regression Testing?

- Testing activities occur after software changes.
- Regression testing usually refers to testing activities during software maintenance phase.

Major testing objectives:

- ***Retest changed components (or parts)***
- ***Check the affected parts (or components)***

Regression testing at different levels:

- ***Regression testing at the unit level***
- ***Re-integration***
- ***Regression testing at the function level***
- ***Regression testing at the system level***

What is Software Regression Testing?

Who perform software Regression:

Developers - regression testing at the unit level or integration
Test engineers - regression testing at the function level
QA and test engineers - regression testing at the system level

What do you need to perform software regression testing?

- Software change information (change notes).
- Updated software REQ and Design specifications, and user manuals.

- Software regression testing process and strategy.
- Software regression testing methods and criteria.

Problems and Challenges in Software Regression Testing

Major problems in software regression testing:

- How to identify software changes in a systematic way?

- ***REQ. specification changes***
- ***Design specification changes***
- ***Implementation (or source code) changes***
- ***User manual changes***
- ***Environment or technology changes***
- ***Test changes***

- How to identify software change impacts in a systematic way?

- ***REQ impacts***
- ***Design impacts***
- ***Implementation impacts***
- ***User impacts***
- ***Test impacts***

Problems and Challenges in Software Regression Testing

Major regression testing problems:

- How to use a systematic method or tool to identify changed software parts?

- How to use a systematic method or tool to identify software change impacts?

- How to use a systematic method or tool to identify affected software test cases?

- How to reduce the re-test suites?

- How to select the test cases in a test suite?

Major challenge in software regression testing:

- How to minimize re-testing efforts, and achieve the adequate testing coverage?

Software Regression Process

Software Regression Process:

Step #1: Software Change Analysis
- *Understand and analyze various software changes.*

Step #2: Software Change Impact Analysis
- *Understand and analyze software change impacts*

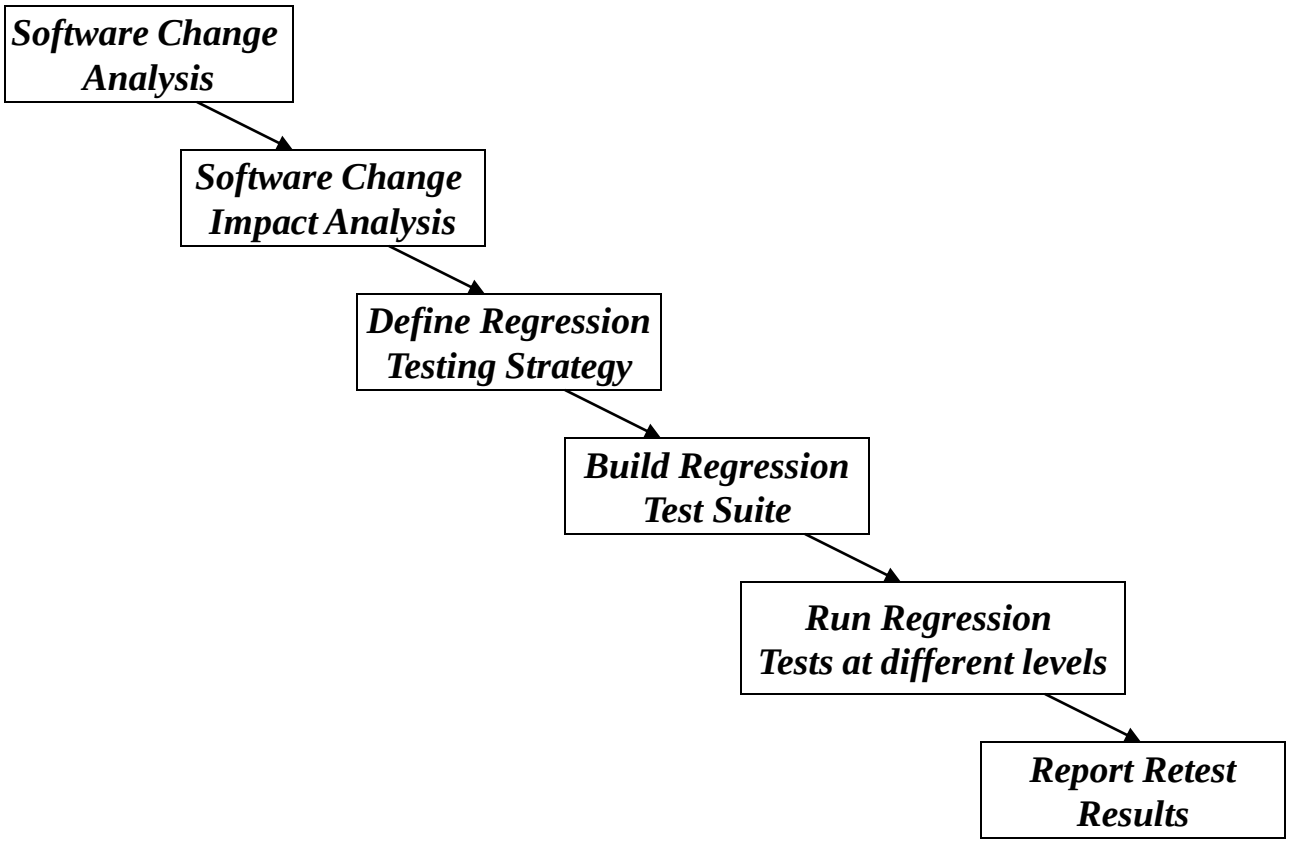
Step #3: Define Regression Test Strategy and Criteria

Step #4: Define, select, and reuse test cases to form a regression test suite

Step #5: Perform re-testing at the different levels.
- *re-testing at the unit level*
- *re-testing at integration level*
- *re-testing at the function level*
- *re-testing at the system level*

Step #6: Report and analyze regression test results

Software Regression Process



Different Types of Software Changes

Requirements analysis	***Requirements Spec. Changes*** ***-> add new functional features*** ***-> change current function features*** ***-> delete existing function features***
System Design	***System architecture changes*** ***-> change component interactions*** ***-> add new components/subsystems*** ***-> update existing components*** ***-> delete existing components***
	High-level design doc. Changes ***-> change state-based behaviors*** ***-> change component interfaces*** ***-> change database design*** ***-> change GUI design*** ***-> change function design***

Different Types of Software Changes

System Design Changes	***- Low-level design doc. Changes*** ***-> change algorithm logic*** ***-> change component structure***
- System implementation	***- Component changes*** ***- internal data types and names*** ***- internal structures, such as*** ***--> class relationships*** ***--> control flow and data flow*** ***- internal functions***
	- Component interface changes ***- call signatures*** ***- message interactions*** ***- protocol messages and formats***
	- Technology and/or language changes

Software Changes Impacts

Types of system changes	*Types of product impacts*
Requirements changes	*Affect design, coding, and testing* *Document update*
Design changes	*Affect coding and tests* *Affect associated components* *Affect system architecture* *Affect related component interactions*
Implementation changes	*Affect test cases, test data, test scripts* *Affect test specification.* *Code change impact s*
Test changes	*Affect other tests.* *Affect test documentation*
Document changes	*Affect other documents.*

Software Regression Strategy

What is a software Regression strategy?

Software test strategy provides the basic strategy and guidelines to test engineers to perform software regression testing activities in a rational way.

Software Regression strategy usually refers to

--> a rational way to define regression testing scope, coverage criteria, re-testing sequence (or order) and re-integration orders.

Software regression test models are needed to support the definition of software regression test strategy, test cases, and coverage criteria.

Typical regression test models:

control flow graph, state-based behavior diagram
object-oriented class diagram
scenario-based model
component-based Regression model

Traditional Software Regression Strategy Based on The Firewall Concept

A Module-Based Firewall Concept for Software Regression Testing:

> ***A module firewall in a program refers to a changed software module and a closure of all possible affected modules and related integration links in a program based on a control-flow graph.***

With this firewall concept, we can reduce the software regression testing to a smaller scope -->

> ***All modules and related integration links inside the firewall.***

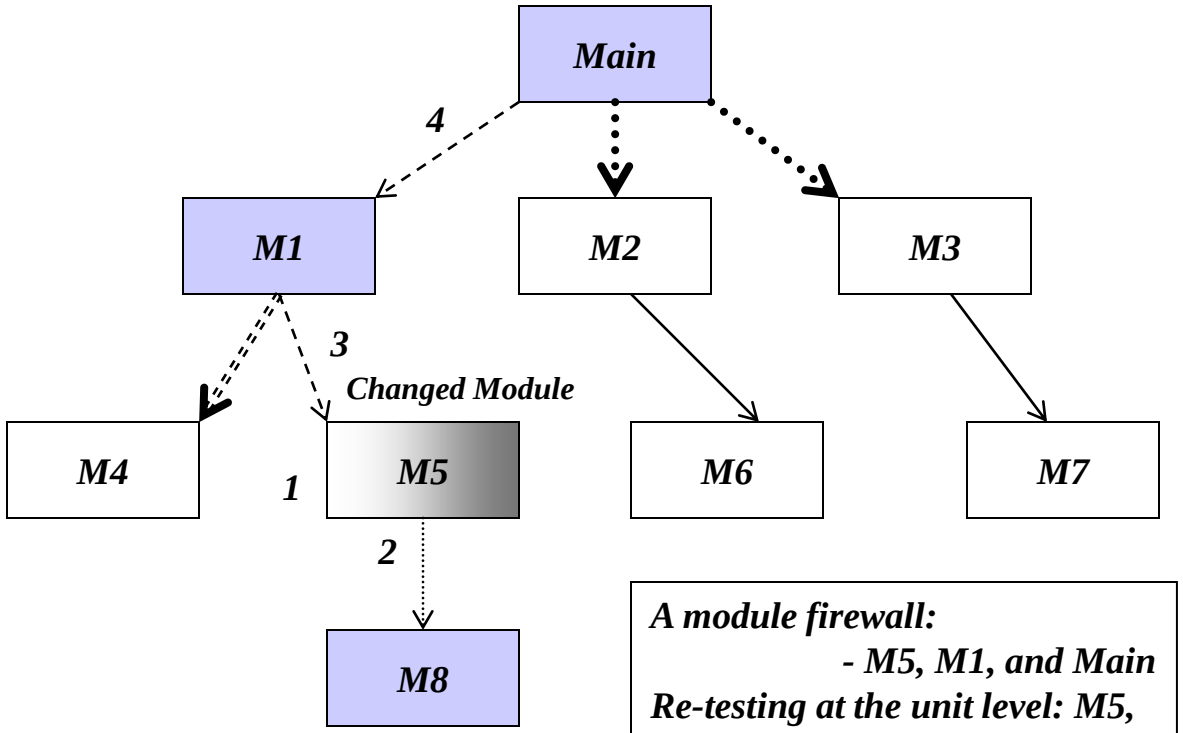
This implies that:

- ***re-test of the changed module and its affected modules***
- ***re-integration for all of related integration links***

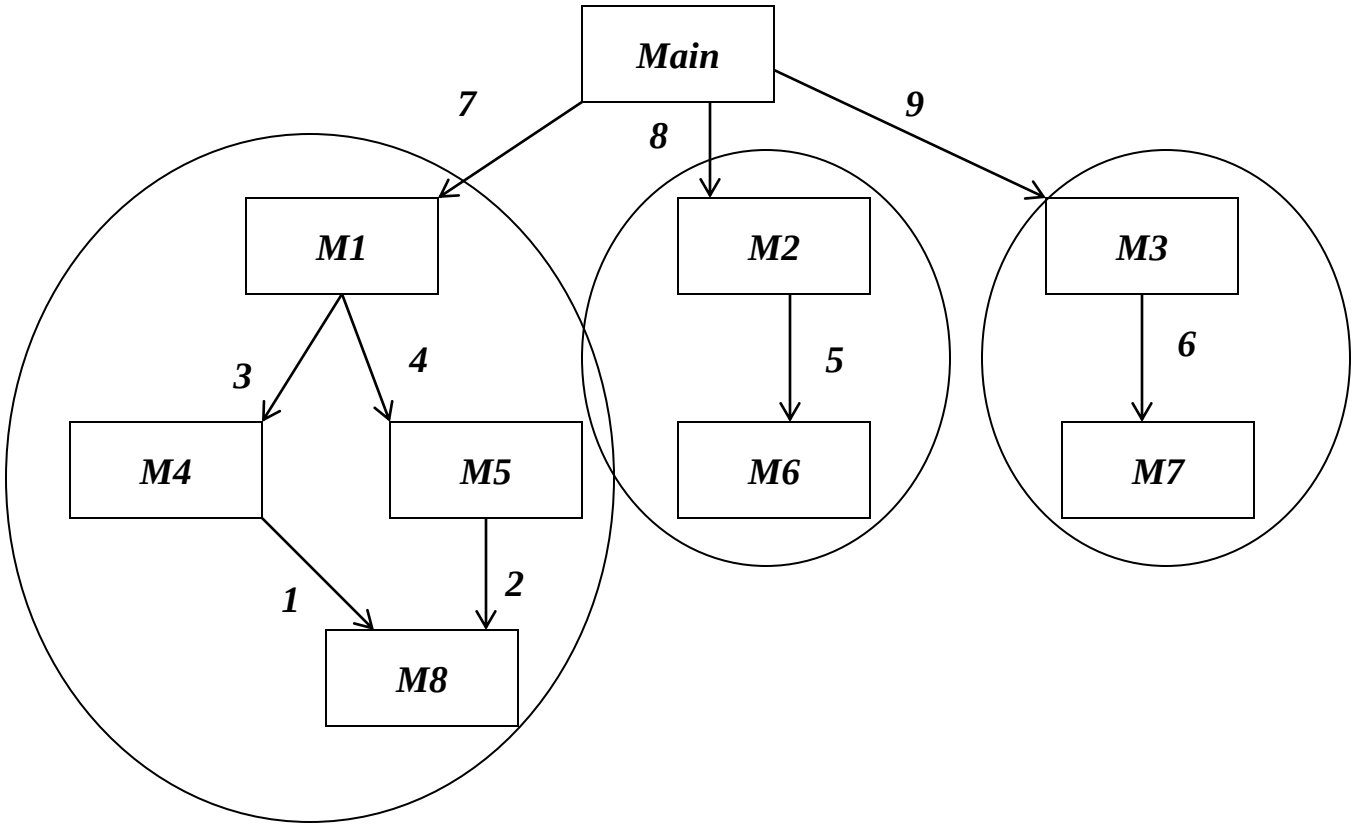
Similarly, we can come out different kinds of firewalls based on various test models.

- ***data firewall,*** ***function firewall***
- ***class firewall,*** ***state/transaction firewall***

A Changed Module Firewall For Regression Testing



Regression Testing for State-Based Behavior Changes



An Object-Oriented Software Regression Strategy

An OO Software Regression Testing Strategy Based on the Class Firewall Concept:

- Identify changed classes

- Identify affected classes using the concept of Class Firewall

- Apply the Class Test Order strategy to classes in a class firewall to perform class re-testing at the unit level

- Use the Class Test Order to re-integrate classes together.

- Select, reuse, and define test cases based on the class firewall and change information.

The Class Firewall Concept

A class firewall concept in OO Software is very useful for OO regression testing.

What is a class firewall?

- A class firewall is a closure set of all classes that are directly or indirectly dependent on the changed class in an OO program.

- The class firewall provides the safe scope of regression testing for an OO software after changing a class.

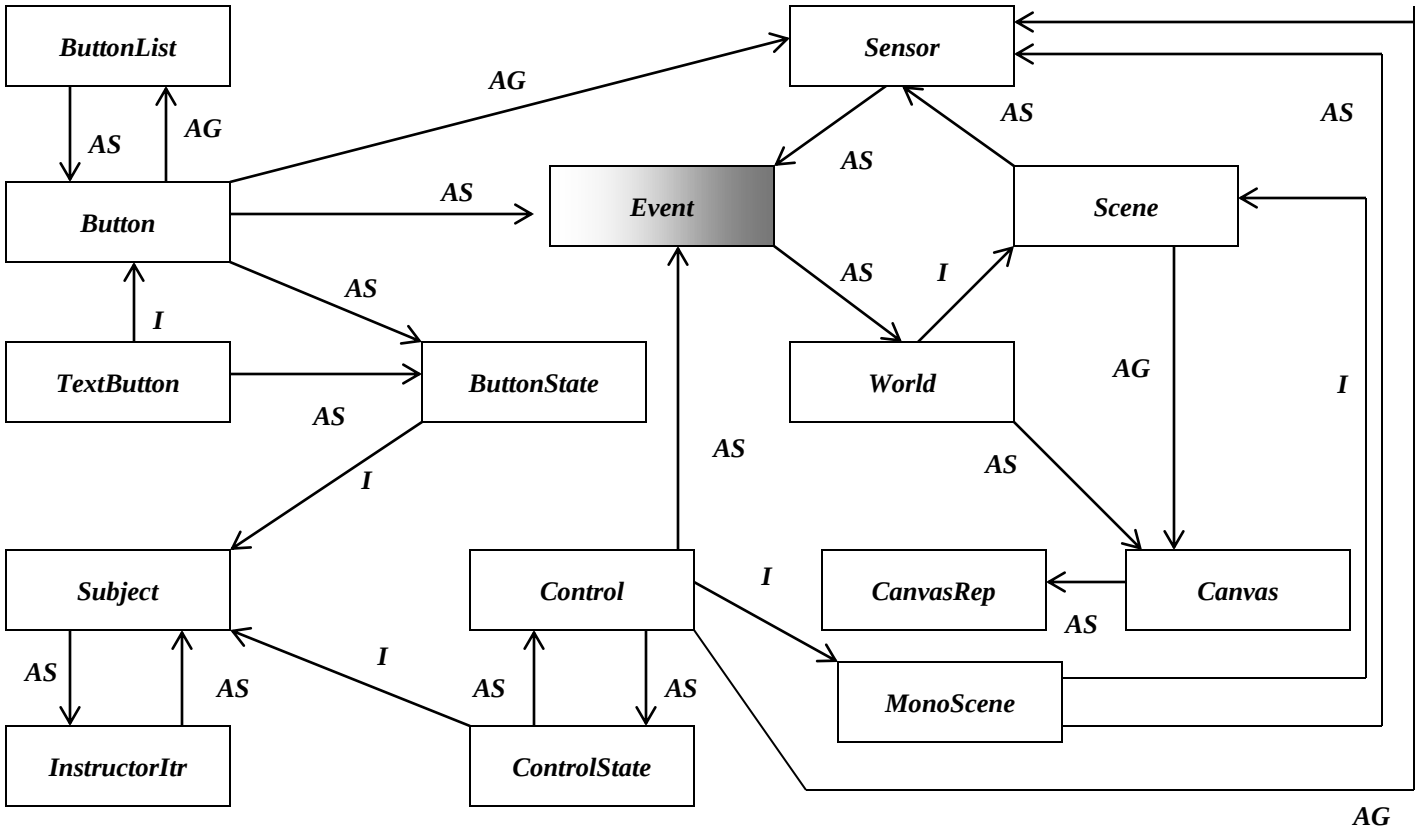
- Similarly, we can apply to many changed classes.

Class Firewall Application:

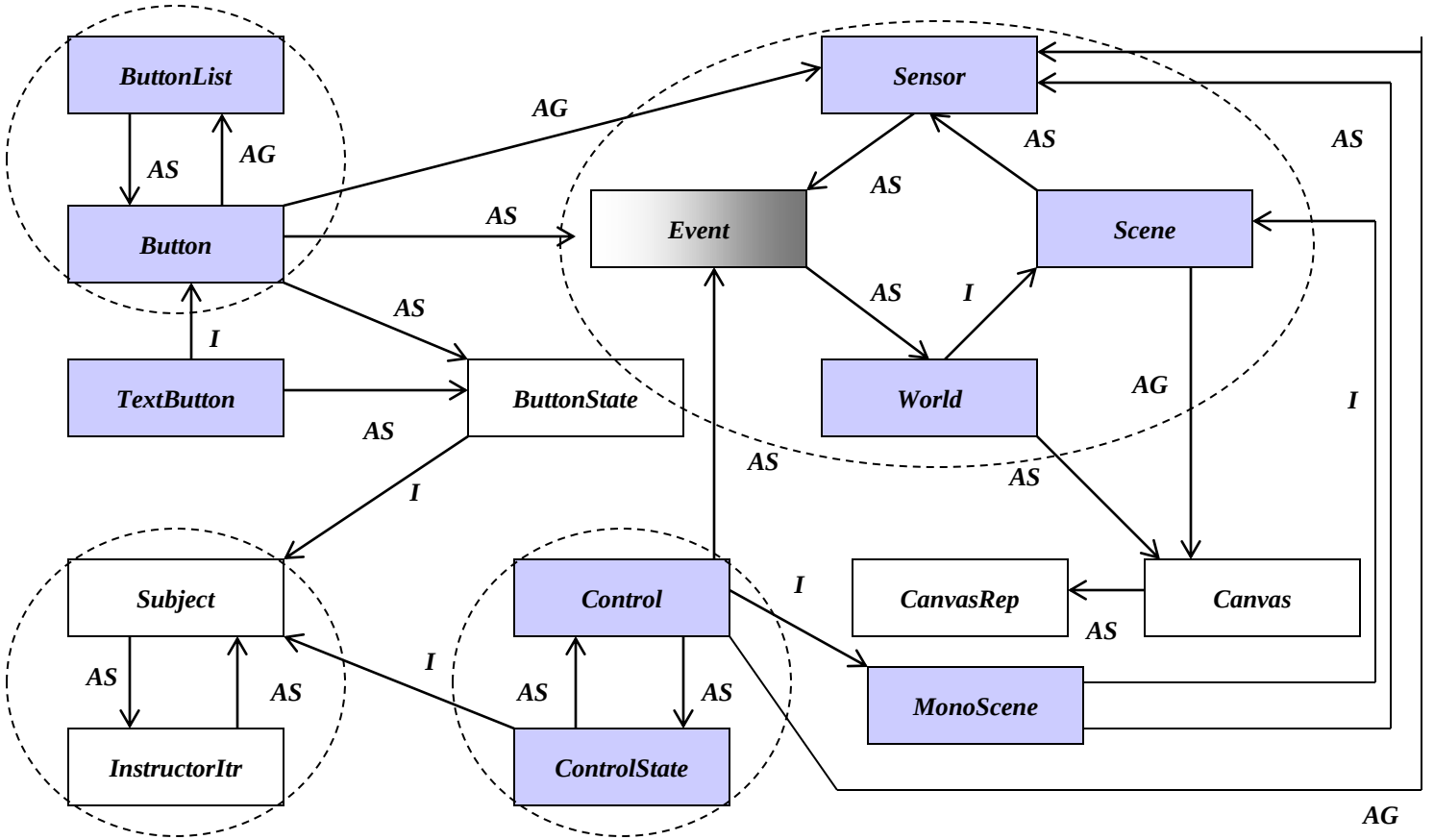
- With this class firewall concept, we can narrow down the class regression testing scope, including unit re-testing, and re-integration.

- Based on the class firewall and changed information, we can select, define, and reuse class test cases for regression testing.

A Changed Class in An Class Relation Diagram



A Class Firewall for An Object-Oriented Programs



A Class Test Order for The Class Firewall