

# of unit tests = 30  
# of integration tests = 0

---



# Continuous Testing I

---

## Outline

- Today we look at the role of testing in **software maintenance**, and the need for **continuous** testing methods
- We'll look at :
  - software maintenance and evolution
    - **corrective**, **adaptive**, **perfective** and **preventive** maintenance
  - continuous testing methods
    - maintaining **functionality**, **failure**, **operational** test suites
    - regression testing

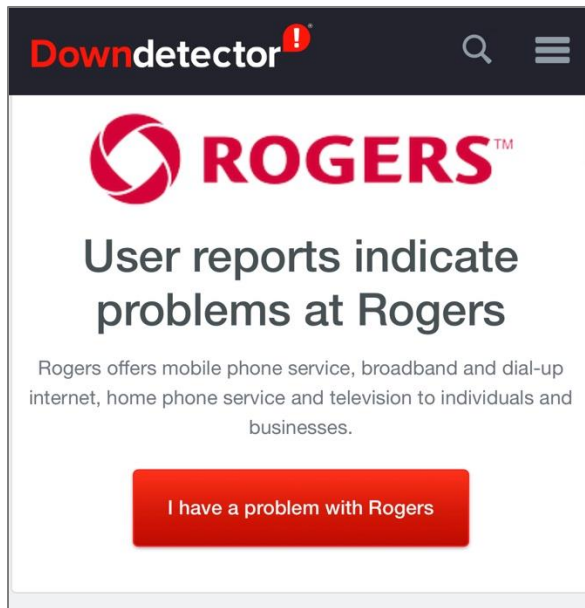
# Software Maintenance

---

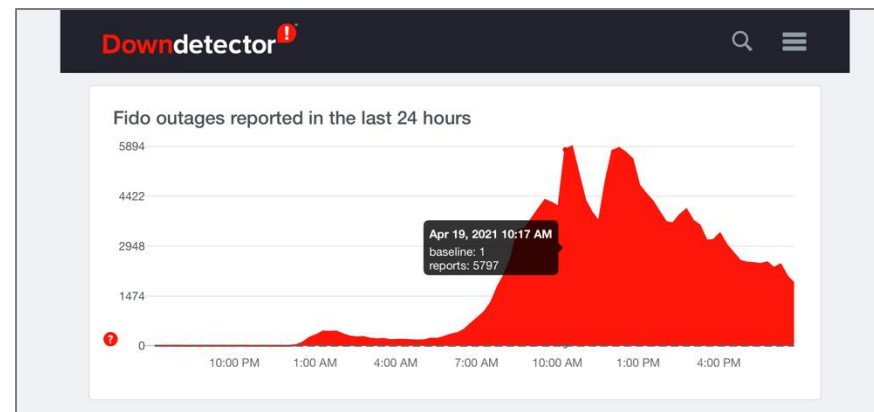
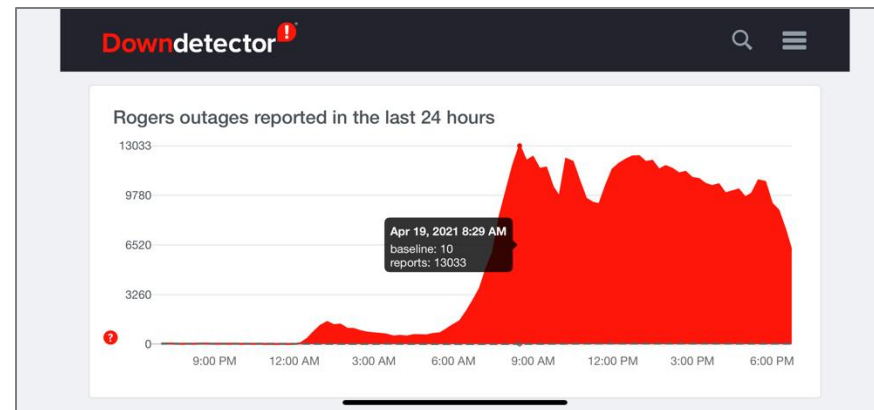
## Evolution

- Maintenance is the phase of development in which the software is in **production** day to day by **real users**
- For successful software, this is almost all of its lifetime, and the software **evolves** in response to observed failures and new requirements
- Usual estimate is that **over 85%** of the total software effort is in maintenance
- Four kinds of software maintenance: **corrective**, **adaptive**, **perfective** and **preventive**

# Case Study: Rogers Wireless Outage (2021)



The image shows the DownDetector website for Rogers. At the top, the DownDetector logo is on the left, and search and menu icons are on the right. Below the logo is the Rogers logo. The main heading reads "User reports indicate problems at Rogers". Underneath, a paragraph states: "Rogers offers mobile phone service, broadband and dial-up internet, home phone service and television to individuals and businesses." At the bottom, there is a red button that says "I have a problem with Rogers".



The image displays two tweets from CBC News Alerts (@CBCAlerts). The top tweet, posted 1 day ago, reports that Rogers is experiencing voice and data outages across Canada, with the company promising frequent updates. It includes a photo of a Rogers store entrance. The bottom tweet, posted 19 hours ago, provides an update stating that Rogers has identified the cause of the Canada-wide outage but has not yet provided a time for service return, noting that the outage affects many customers coast-to-coast, including some 911 and fire services. Both tweets show engagement metrics like replies, retweets, and likes.

# Case Study: Rogers Wireless Outage (2021)





# Case Study: Rogers



**ROGERS**What We DoLife at RogersGiving BackOur Impact60 ProjectRogers & ShawNews & IdeasEN FR

[← Back to all news](#)

To our valued customers,

Especially during these times, we know how important it is to stay connected and how much you rely on our services for work, school and staying in touch.

The intermittent wireless service issues that started earlier this morning are unacceptable. On behalf of all of us here at Rogers, Rogers for Business, Fido, and chatr, I want to sincerely apologize for the significant impact and frustration that this has caused.

Our team of network experts, alongside our network partner Ericsson, are working hard to restore full service and have identified the root cause of the issue to help ensure it doesn't happen again.

This situation is continuing to evolve, and I wanted to share what we know so far:

**When did this start?**

Early this morning, our network operations centre started to see that some wireless customers were experiencing intermittent issues with voice calls, SMS and data services. Our TV, home and business wireline Internet, and home phone services were not impacted.

**What happened?**

We have identified the root cause of the service issues and pinpointed a recent Ericsson software update that affected a piece of equipment in the central part of our wireless network. That led to intermittent congestion and service impacts for many customers across the country.

**What are you doing about it?**

We have addressed the software issue and our engineering and technical teams will continue to work around the clock with the Ericsson team to restore full services for our customers.

**When will services be restored?**

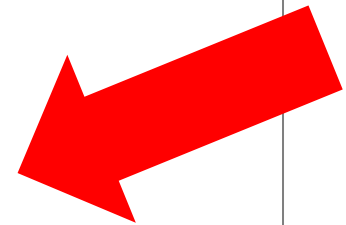
We do not have an exact time yet as it may take us several hours to get everything back up and running normally. It's important that we bring wireless services back up gradually as we return to full service. You have our full commitment that we will not rest until all services are restored.

**How can I be updated?**

We will continue to provide updates every few hours. Please visit Rogers.com or any of our social media channels for the most up to date information.

Sincerely,

Jorge Fernandes  
Chief Technology Officer  
Rogers Communications



# Corrective Maintenance

---

## Kinds of Errors

- **Corrective** maintenance is concerned with fixing reported failures (errors) observed in the software
- These can themselves come in three varieties:
  - Coding errors
    - typically easy and inexpensive to correct since they are confined within one unit
  - Design errors
    - more expensive since they may involve changes to several units
  - Requirements errors
    - most expensive since they often involve extensive system redesign (re-architecting) to correct
- Often a short-term fix is implemented first until a more permanent change can be made later

# Adaptive Maintenance

---

## New Environments

- Adaptive maintenance:
  - involves changing the software to work in some new environment such as a different machine/hardware or operating system
  - involves changing one part of a software system due to changes in another component or subsystem
- Characterized by no change in functionality, just a secondary change due to other parts of the system evolving or a new environment .



# Perfective Maintenance

---

## Improving Performance and more...

- **Perfective** maintenance includes changes made to improve the performance, maintainability, etc. of the program
- It does not always occur as the result of failures or faults in the source code.
  - e.g., improving architecture to enhance performance
  - e.g., improving documentation or code formatting to increase readability

# Preventive Maintenance

---

## Changes to Prevent Failure

- **Preventive** maintenance involves changes that may result from faults uncovered by a programmer who addresses the fault so that it does not become a failure.  
e.g., better exception handling in a Java program

# Maintenance Tests

---

## Maintaining Quality

- As time goes on, the software is often maintained by programmers not involved in the initial design and development
  - These programmers tend to be more focused on the **changes** than the **whole product**
- For this reason, testing has an even **more important** role in quality assurance in the maintenance phase than it does in initial development and delivery
  - it helps to make sure that changes have not broken anything

# Continuous Testing Methods

---

## Testing as a Maintenance Activity

- Thus, testing is not a one-time thing – we’re never “done” testing
- As software is maintained, if we are to maintain consistent quality, we must continue testing - both of old existing functionality, and of new introduced functionality
- For this reason, Agile calls for continuous testing
  - test every version, every day
- At a minimum, we must re-test thoroughly after every set of changes, before the changed software is released

# Test Suites

---

## Tests Are Part of the Product

- Most projects maintain **test suites**, sets of tests to be run on every release of the software
- Maintained in **parallel** with the software - often with **at least** as much effort as the software itself
  - as we have already seen, **automation** is essential to make this practical

## Kinds of Test Suites

- Three (related) kinds of continuous tests are normally performed and maintained continuously in software maintenance
  - **Functionality** (or acceptance) testing, **failure** testing, **operational** testing



# Continuous Functionality Testing

---

## Functionality Suites

- We have already seen **functionality** and **acceptance** testing suites (you've built one!)
- When used **continuously** over the evolution of the software, we **maintain** the functionality tests by:
  - Every time a **new feature** is added, **new tests** specifically aimed at testing that feature are permanently added to the test suite  
(In Agile we **must** have these new tests since they form the **specification** for the new software capabilities)
  - Every time a **feature** is changed or extended, we change or extend the corresponding functionality **tests** to correspond

# Failure Testing

---

## Failure Suites

- **Failure tests** are suites of examples that have been observed to cause a failure of the software in the past
- To be effective, failure tests must be **maintained** over the evolution of the software, by:
  - Before correcting any observed failure, characterize the failure by creating a "**failure test**" that causes it
  - This becomes the **specification** of the fix - the changed software must at a minimum correct the error for the test
  - The failure test must **cause** the error in the old software and **not cause** the error in the new software
- We keep all such tests in a **failure test suite** to be re-run on all future versions of the software, to ensure that the failure does not **reappear** due to a future change

# Operational Testing

---

## There's No Substitute for the Real Thing

- Operational tests are suites of tests that are actual production runs observed in the use of the software
  - e.g., for our Banking **back end**, all of the banking transaction files from a set of **real front ends** run all day on a particular day

# Operational Testing

---

## There's No Substitute for the Real Thing

- Operational test suites must be created **early** in the **production life** of the software by sampling actual production runs  
e.g., instrumenting bank machine software to capture the actual transactions from customers over a day
- Must be **updated** to add new real operational tests each time significant new or changed features are added to the software
- These tests form a sort of **sanity check** on the software...  
...to make sure that when we are about to release a new version, it will not only still run our artificial tests but will also still handle real customer input (could be **embarrassing** otherwise)

# Regression Testing

---

## Comprehensive Continuous Testing

- **Regression testing** refers to an automated continuous testing strategy, whose purpose is to make sure that the software does not "**regress**" - that is, become less effective than it has been in the past
- Regression test suites are normally **comprehensive**, including (at least) the three components:
  - **Functionality** tests, to make sure that we still meet the basic requirements
  - **Failure** tests, to make sure that we haven't recreated a past failure
  - **Operational** tests, to make sure that we can still process real production runs
- Each of these is maintained, either together or separately, as previously described



# Continuous Testing I

---

## Summary

- Software maintenance, consisting of **corrective**, **adaptive**, **perfective** and **preventive** steps, is the longest phase of software development
- Continuous testing is **essential** to maintain quality during software maintenance
- Regression testing combines **functionality**, **failure** and **operational** testing in an automated continuous testing framework

## Next Time

- Practical regression testing