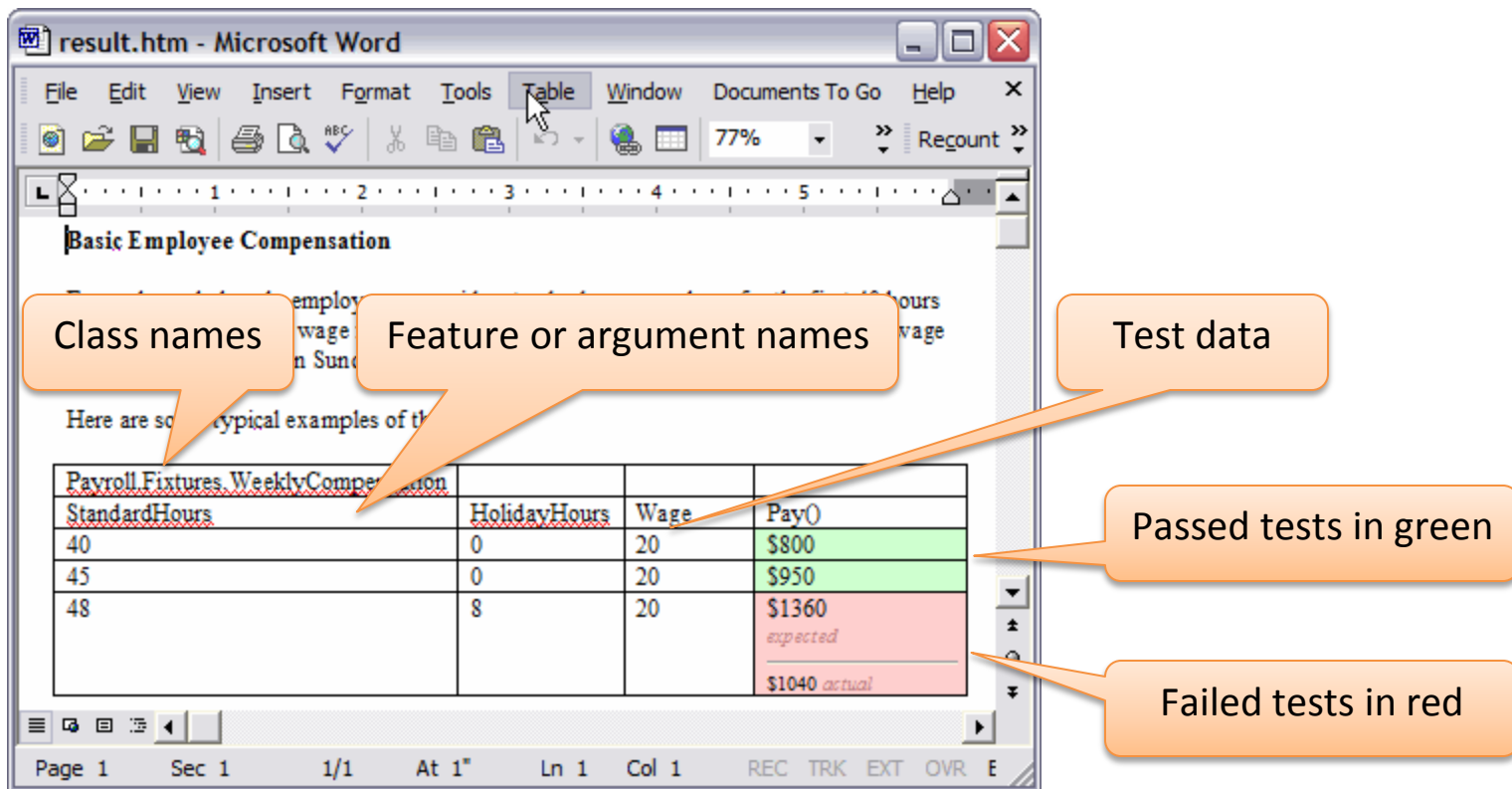


Framework for Integrated Test(Fit)

Requirement for both customers and programmers

Goals:

- Easy for customers to write and validate.
- Easy for programmers to test and verify.



The screenshot shows a Microsoft Word document titled "result.htm" with a table of test data. The table has four columns: "StandardHours", "HolidayHours", "Wage", and "Pay()". The first two rows have green backgrounds, indicating passed tests. The third row has a red background, indicating a failed test. The "Pay()" column for the failed test shows "expected" and "actual" values.

StandardHours	HolidayHours	Wage	Pay()
40	0	20	\$800
45	0	20	\$950
48	8	20	\$1360 <i>expected</i> \$1040 <i>actual</i>

Callouts in the image:

- Class names: `Payroll.Fixtures.WeeklyCompensation`
- Feature or argument names: `StandardHours`, `HolidayHours`, `Wage`
- Test data: `40`, `0`, `20`, `$800`
- Passed tests in green: The first two rows of the table.
- Failed tests in red: The third row of the table.

Another Fit example

Division		
numerator	denominator	quotient()
1000	10	100.0000
-1000	10	-100.0000
1000	7	142.85715
1000	.00001	100000000
4195835	3145729	1.3338196



This is the famous Pentium bug.

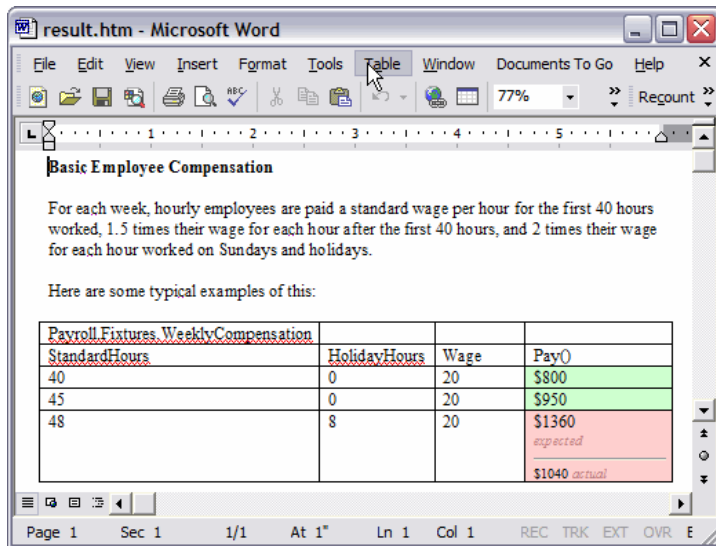
Integration in two levels

- Test development is integrated with specification (**Specification By Example**).
- Test execution is integrated in that test data flows through *fixtures* to the same interfaces developers use while programming.

Further benefit: when a test fails, we immediately know to which part of the requirement the failing test affects.

Fixtures

Fixtures interpret what requirement tables mean and automatically derive test cases from them.



Basic Employee Compensation

For each week, hourly employees are paid a standard wage per hour for the first 40 hours worked, 1.5 times their wage for each hour after the first 40 hours, and 2 times their wage for each hour worked on Sundays and holidays.

Here are some typical examples of this:

StandardHours	HolidayHours	Wage	PayO
40	0	20	\$800
45	0	20	\$950
48	8	20	\$1360 <i>expected</i>
			\$1040 <i>actual</i>

Customers and programmers have to agree on certain naming conventions.

test_pay

local

payroll: PAYROLL

do

create payroll

payroll.set_standard_hours(40)

payroll.set_holiday_hours(0)

payroll.set_wage(20)

check payroll.pay = 800 end

end

Specify action requirements

Actions for browser			
Action	argument		
Start	firefox, opera		
Open	www.google.com		
Goto	text field named q		
Enter	Software Architecture		
Check	result page	first item	"Software Architecture, ETH Zurich..."
Close	Current page		
Undo	close page		
Check	url	last page	

Issues of ActionFixture:

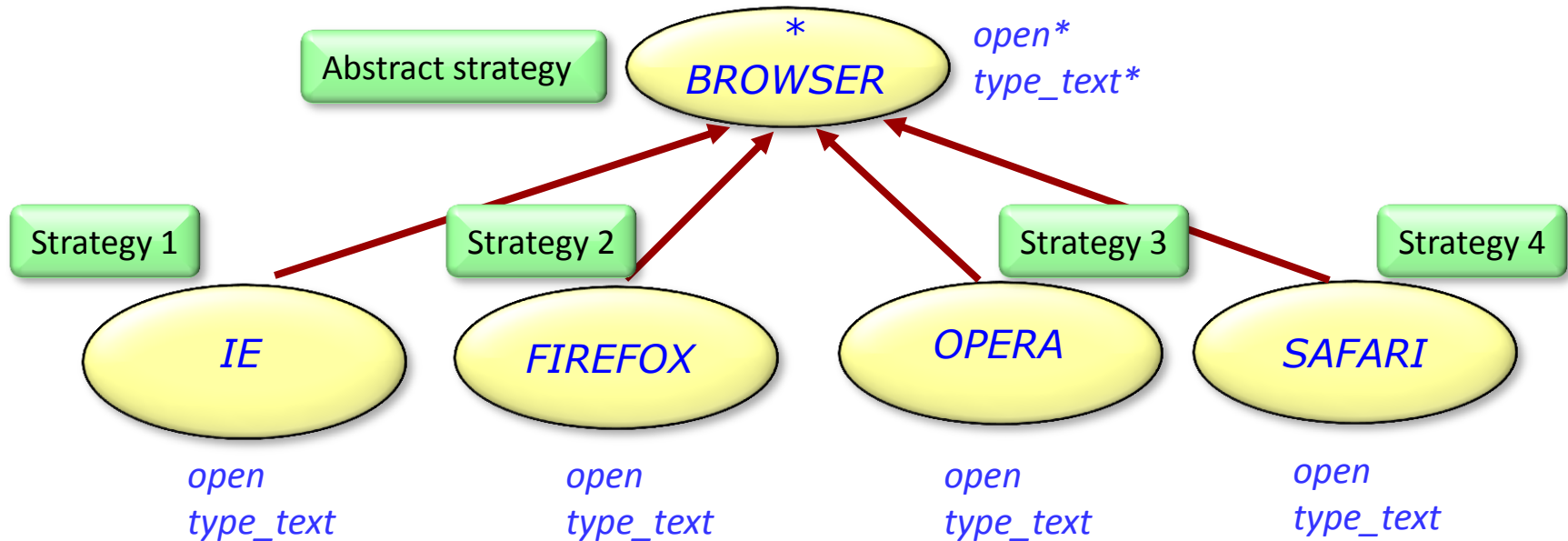
How to support different browsers?

How to perform actions?

How to handle undo-redo?

Handle different browsers

Different browsers with similar interface and usage pattern, such as open a link, enter some text.



Strategy Pattern: different ways to do the same thing

Handle creation

It is nice to have a central place to create all browser instances -- Use **Factory pattern**.

```
class BROWSER_FACTORY
```

```
  feature
```

```
    new_ie: IE do ... end
```

```
    new_firefox: FIREFOX do ... end
```

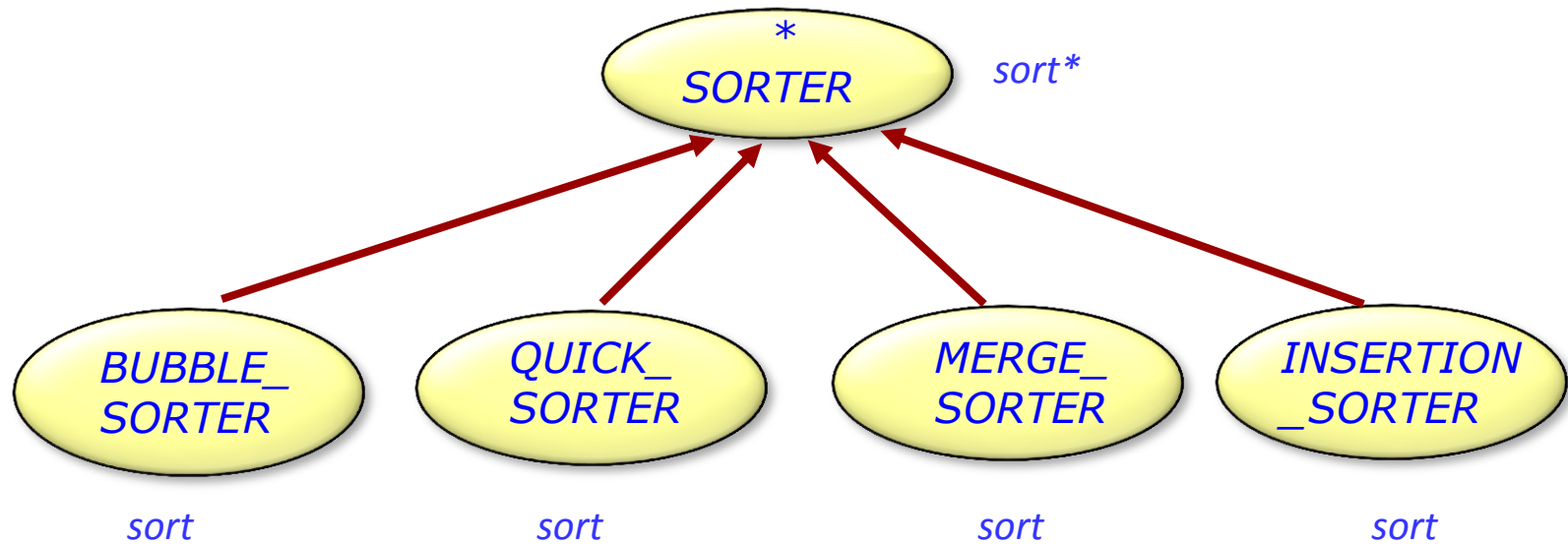
```
    new_opera: OPERA do ... end
```

```
    new_safari: SAFARI do ... end
```

```
end
```


Sorting: another example of strategy pattern

Different sorting algorithms.



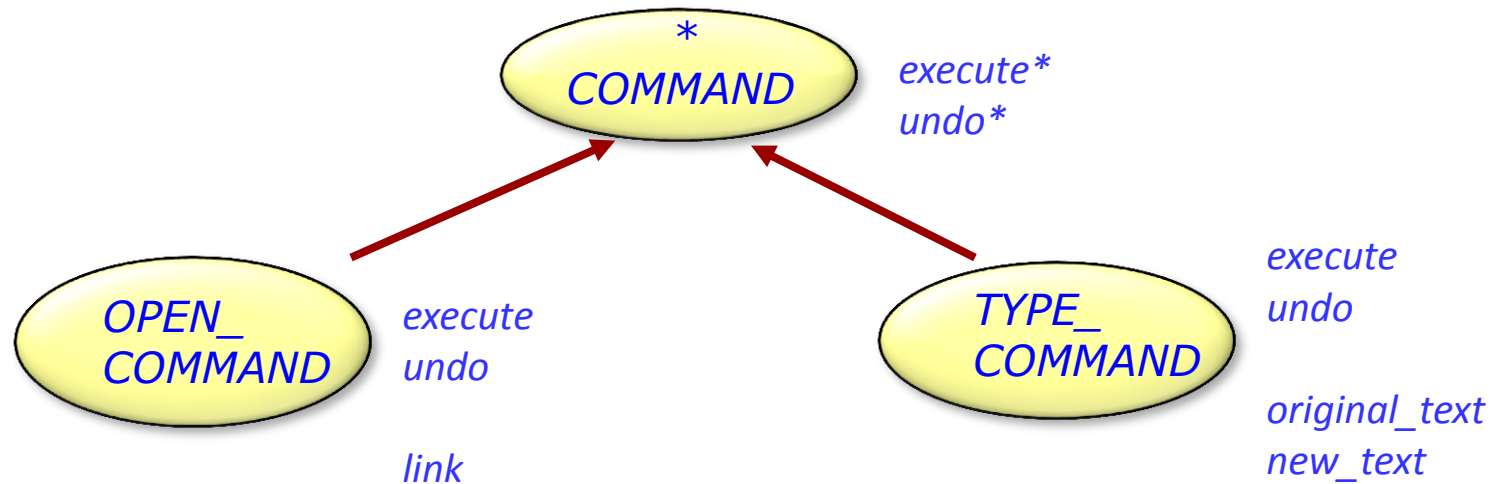
Handle actions

- Every action: execute and undo.
- Different actions require different data:
 - **Open** needs a link name
 - **Enter** needs a piece of text
 - **Enter** also needs to store the original text to support undo.

Actions for browser			
Action	argument		
Open	www.google.com		
Goto	text field named q		
Enter	Software Architecture		
Check	result page	first item	"Software..."
Close	Current page		
Undo	close page		
Check	location	result page	

Command pattern: encapsulate actions

Abstract command class with concrete implementations.



Open needs a *link*, undo of open will close the page for that *link*.

Type needs a *new_text* which is the text to-be-entered. It also needs to keep the *original_text*. Undo will restore the *original_text*.

Let's go back to the Fit framework

result.htm - Microsoft Word

File Edit View Insert Format Tools Table Window Documents To Go Help

Basic Employee Compensation

For each week, hourly employees are paid a standard wage per hour for the first 40 hours worked, 1.5 times their wage for each hour after the first 40 hours, and 2 times their wage for each hour worked on Sundays and holidays.

Here are some typical examples of this:

StandardHours	HolidayHours	Wage	Pay()
40	0	20	\$800
45	0	20	\$950
48	8	20	\$1360

expected
\$1040 actual

Page 1 Sec 1 1/1 At 1" Ln 1 Col 1 REC TRK EXT OVR E

What if the data is not simple numbers, instead, is a complex object, such as a [LINKED_LIST](#) with 2 elements and the first element is a [PERSON](#)?

Fit cannot handle this problem nicely.

One can argue whether a [LINKED_LIST](#) of 2 element is suitable for the level requirement which is designed for sharing knowledge between customers and programmers.

More about Fit

Check <http://fit.c2.com>