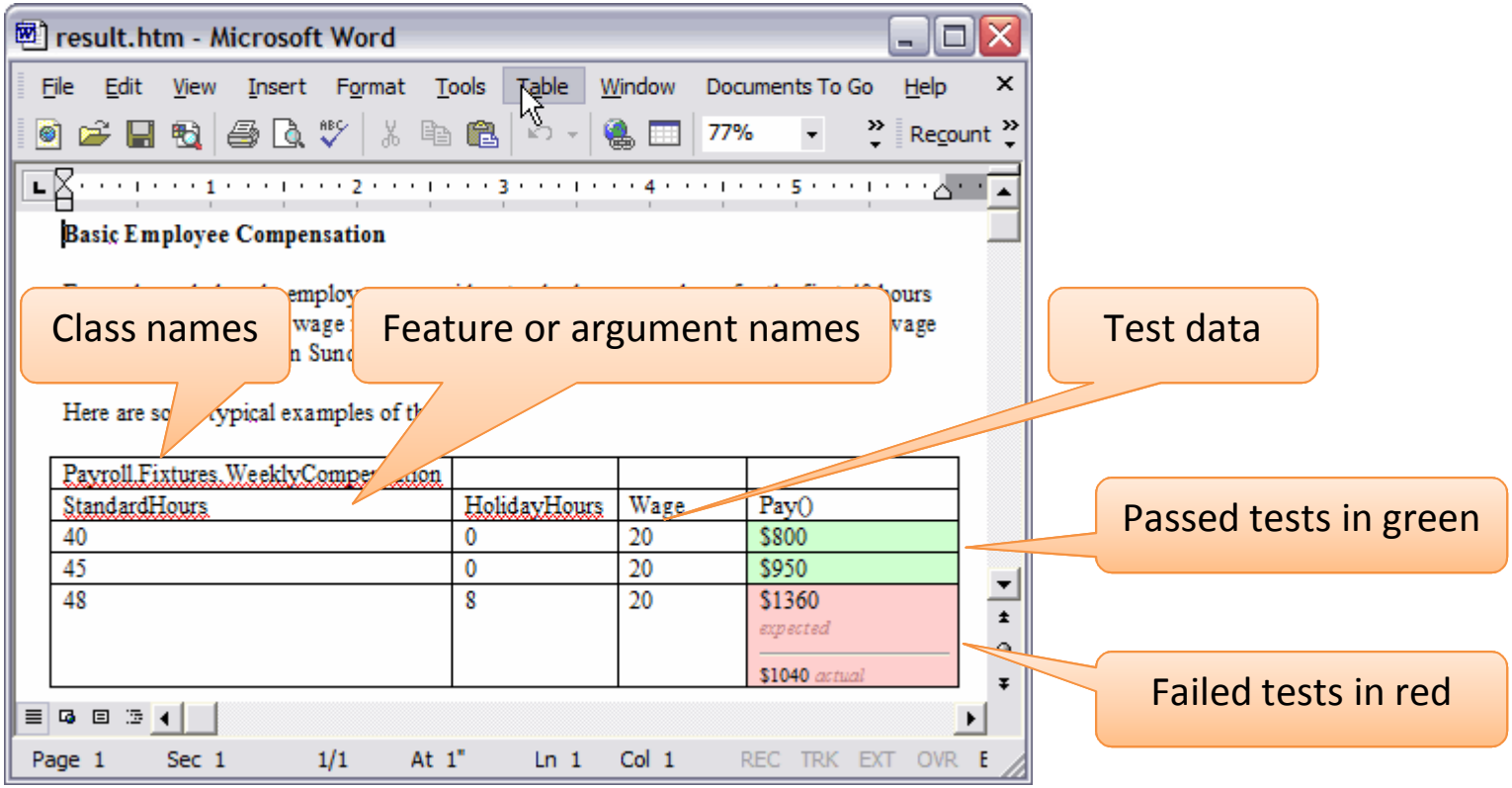
Framework for Integrated Test(Fit)

Requirement for both customers and programmers

Goals:

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



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

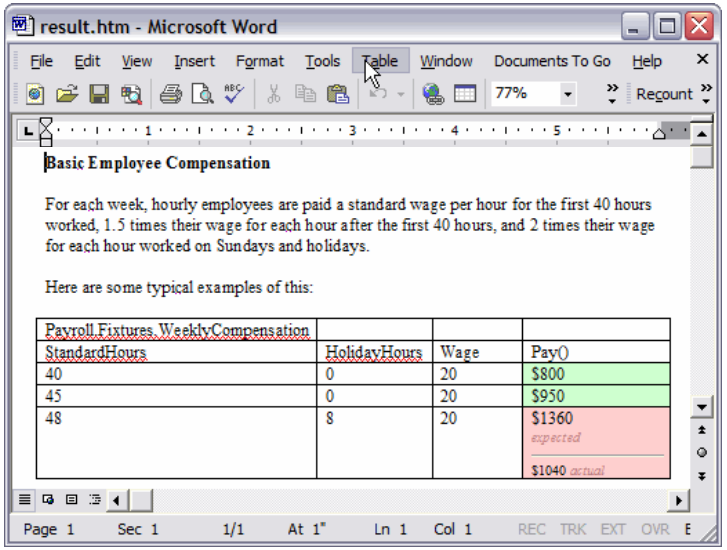
- Test development is integrated with specification (Specification By Example).

- 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

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



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
```

Specify action requirements

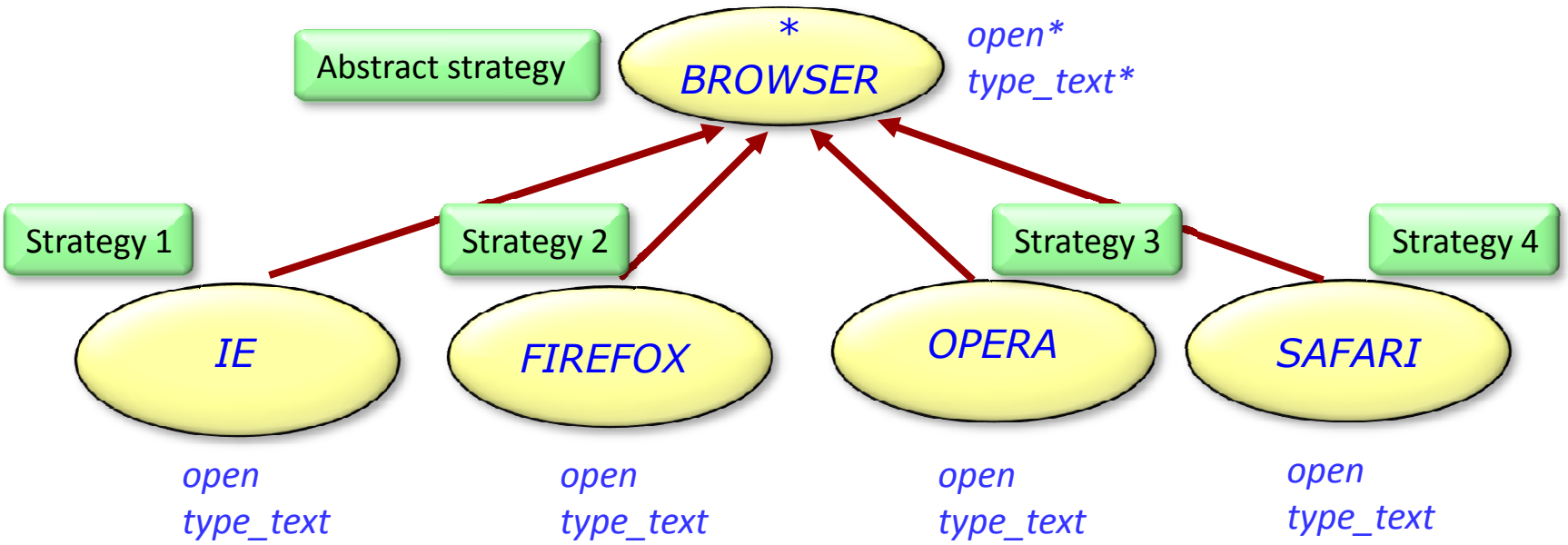
Actions for browser			
Action	**argument**		
Start	firefox, opera		
Open	www.google.com		
Goto	text field named q		
Enter	Software Architecture		
Check	result page	first item	“Software Architecture, ETH Zurich…”
Close	Current page		
Undo	close page		
Check	url	last page	

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.

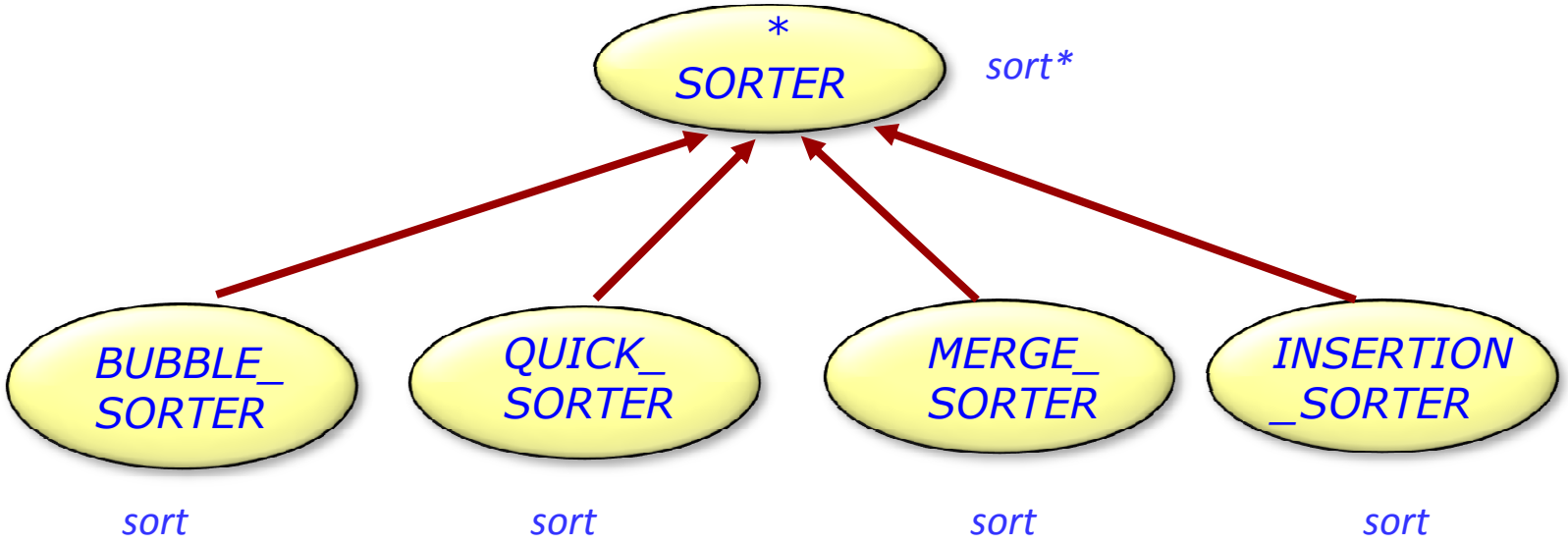
```
class BROWSER_FACTORY

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.



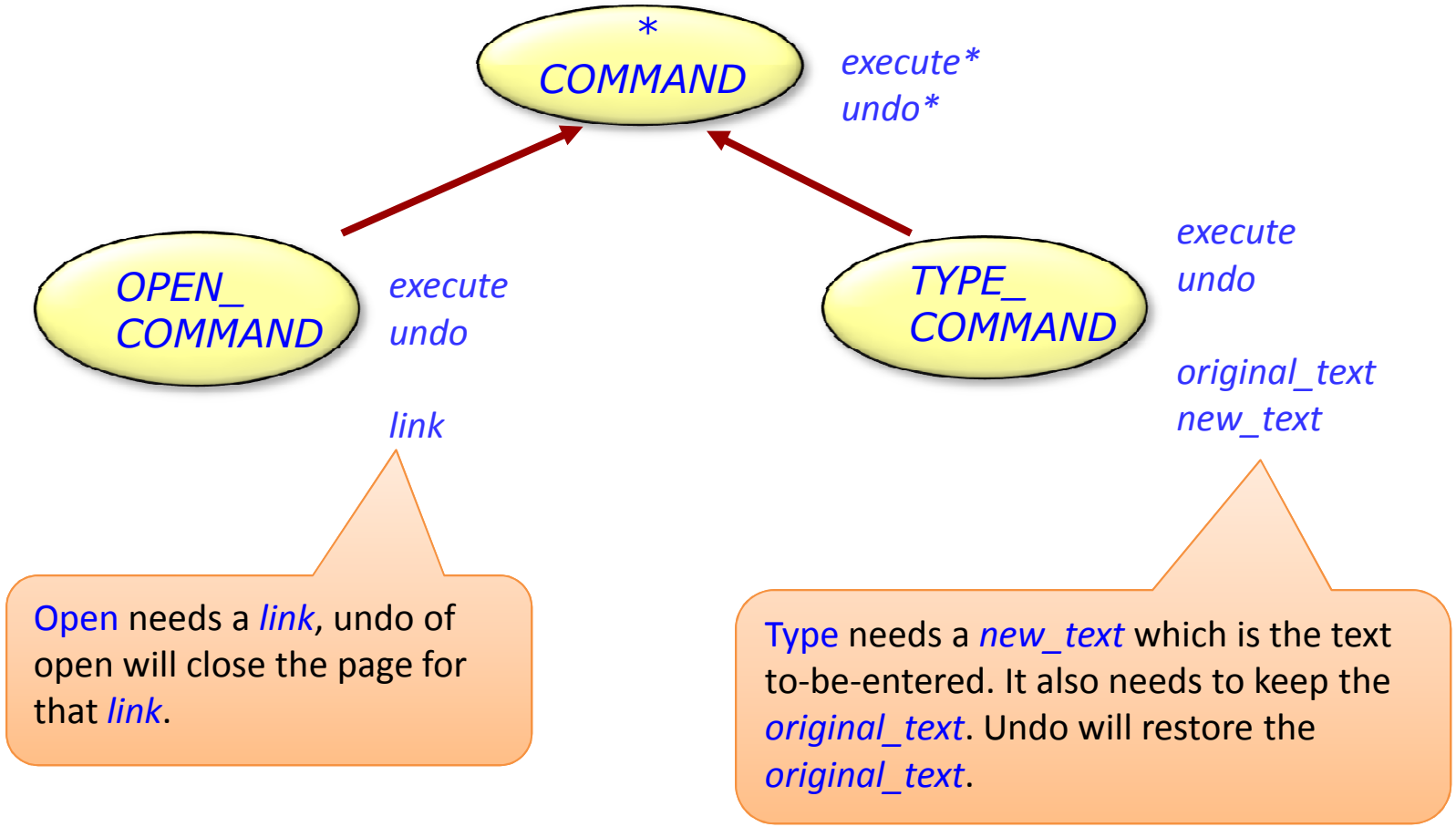
Handle actions

- 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.

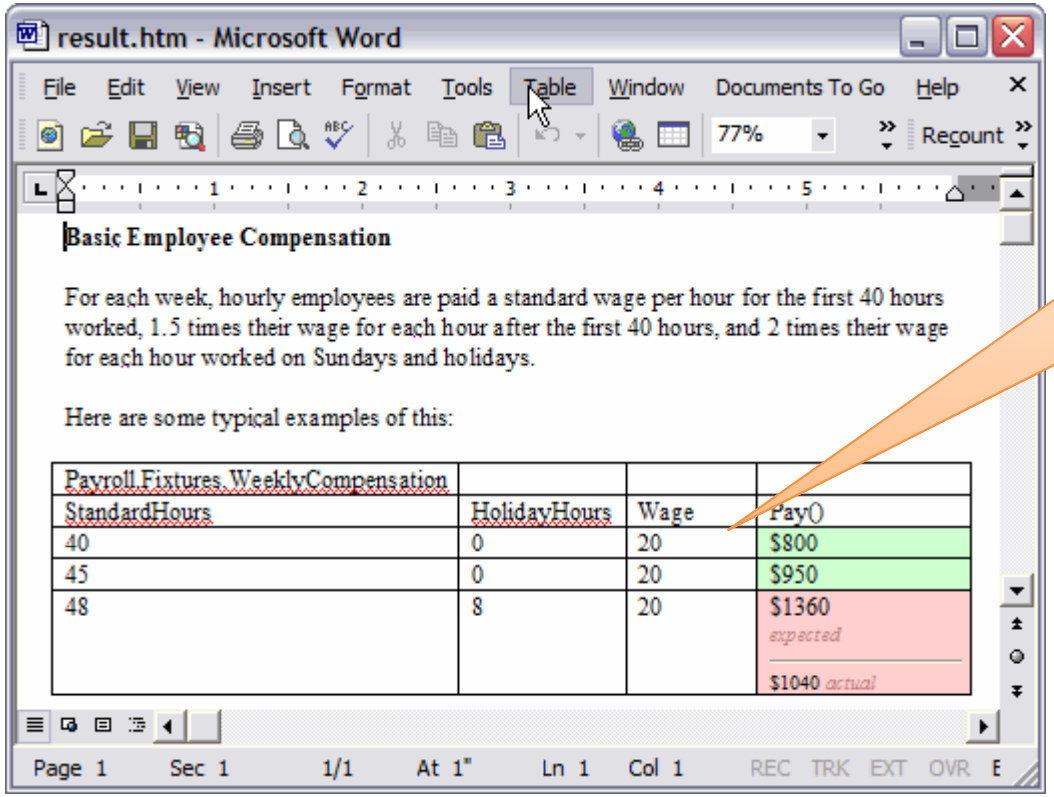
Actions for browser			
Action	**argument**		
Open	www.google.com		
Goto	text field named q		
Enter	Software Architecture		
Check	result page	first item	"Software…"
Close	Current page		
Undo	close page		
Check	location	result page	

Command pattern: encapsulate actions

Abstract command class with concrete implementations.



Let's go back to the Fit framework



What if the data is not simple numbers, instead, is a complex object, such as a LINKED_LIST with 2 element 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