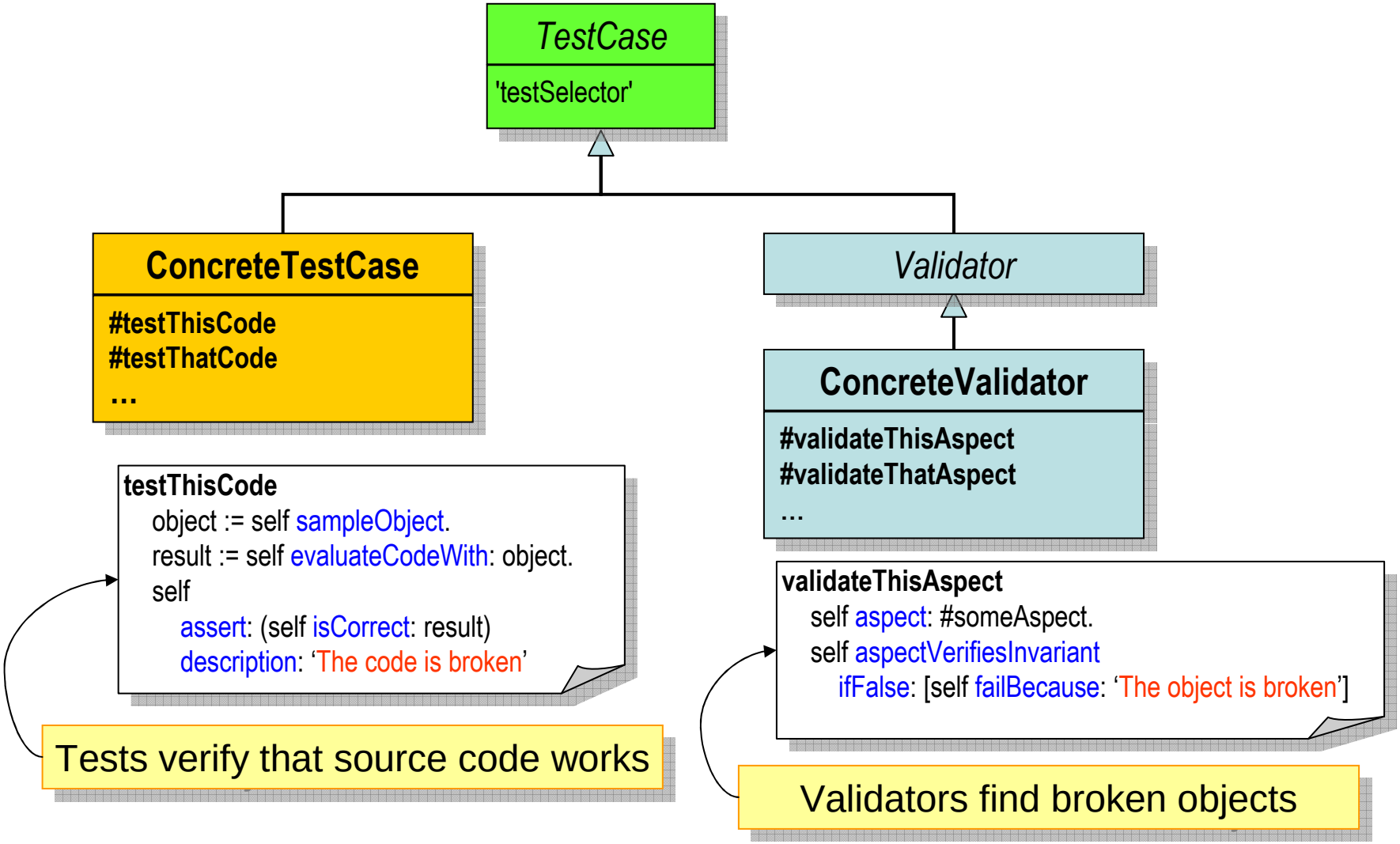
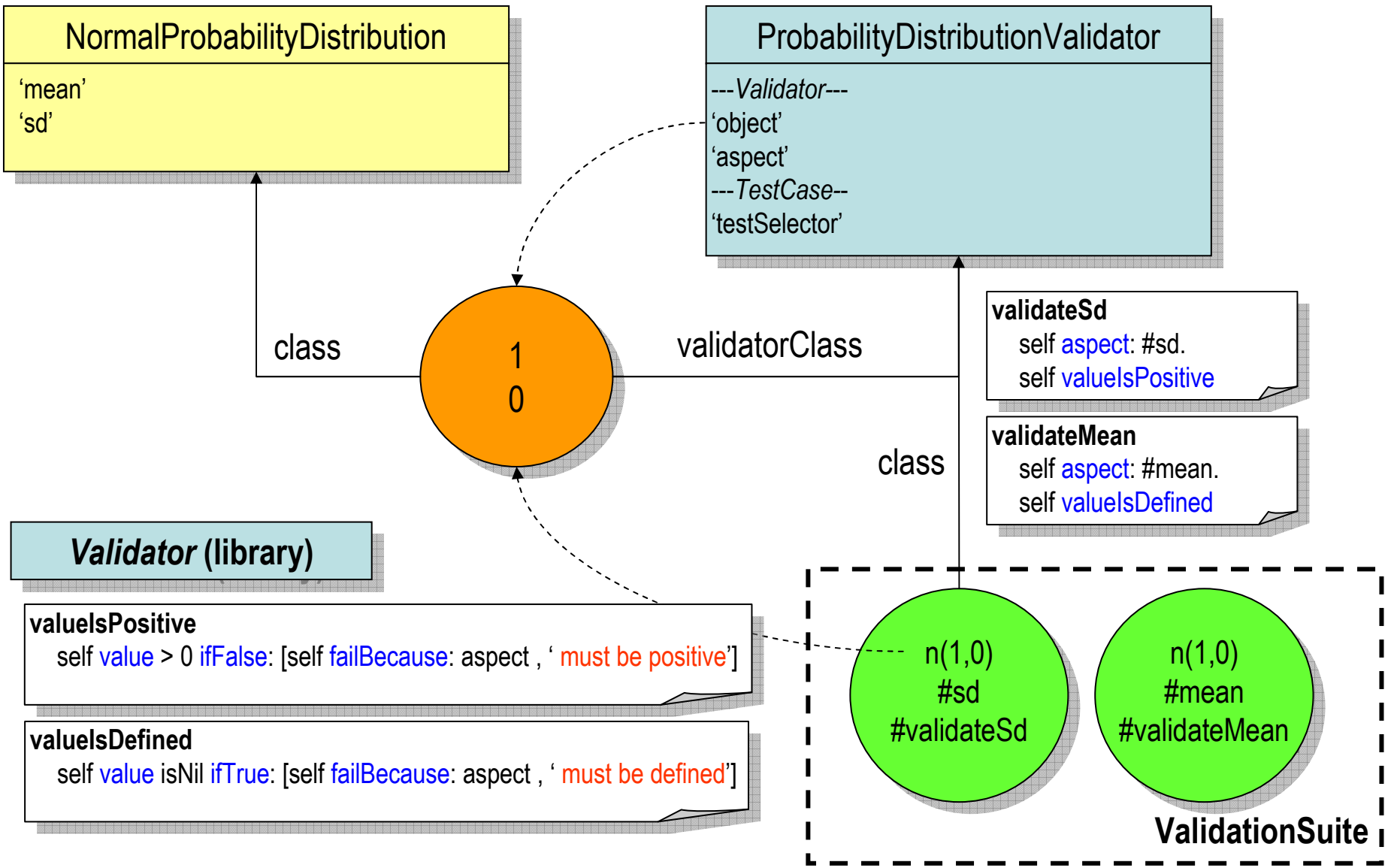
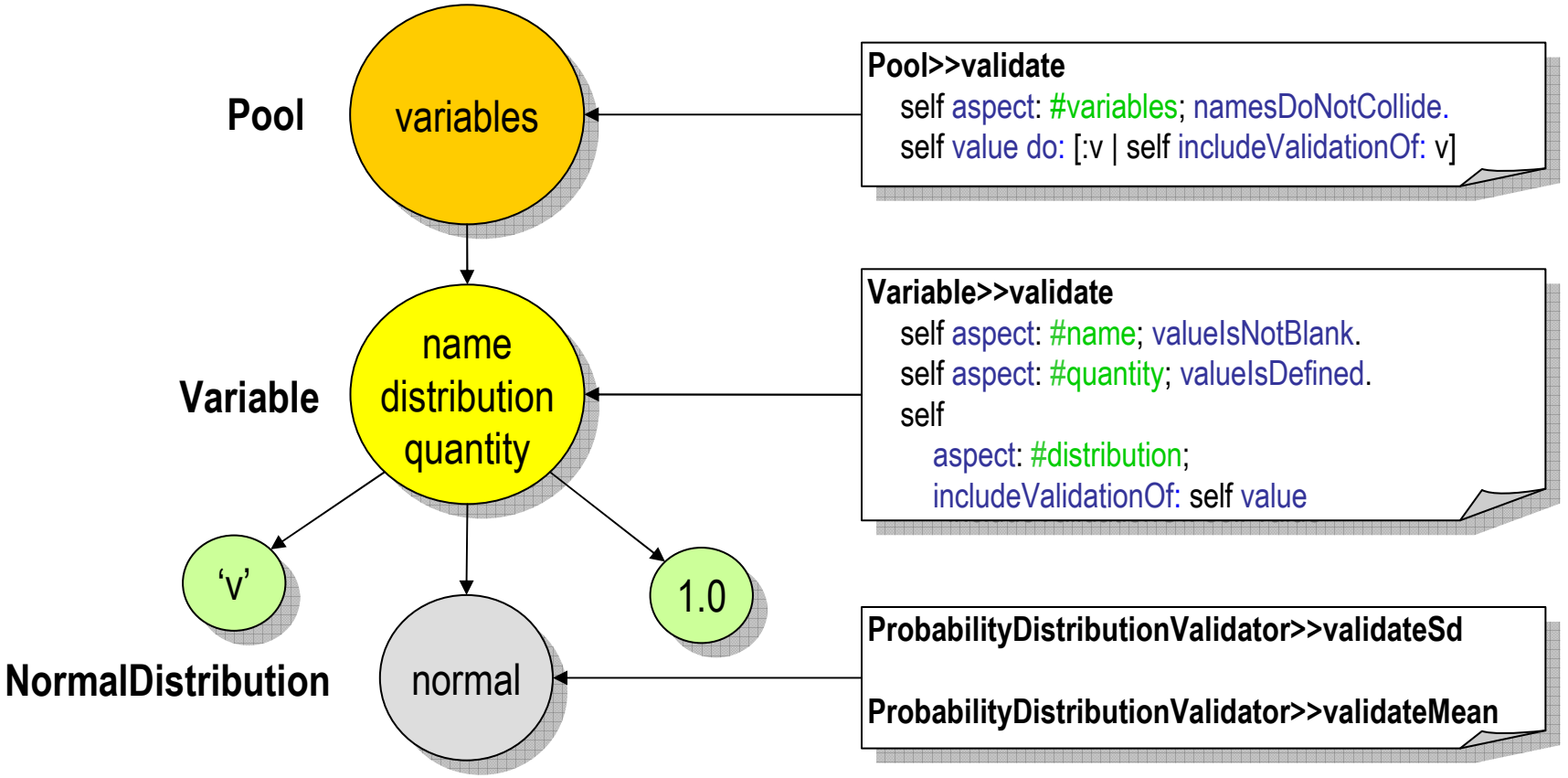
SUnit based Validation



Validating an object



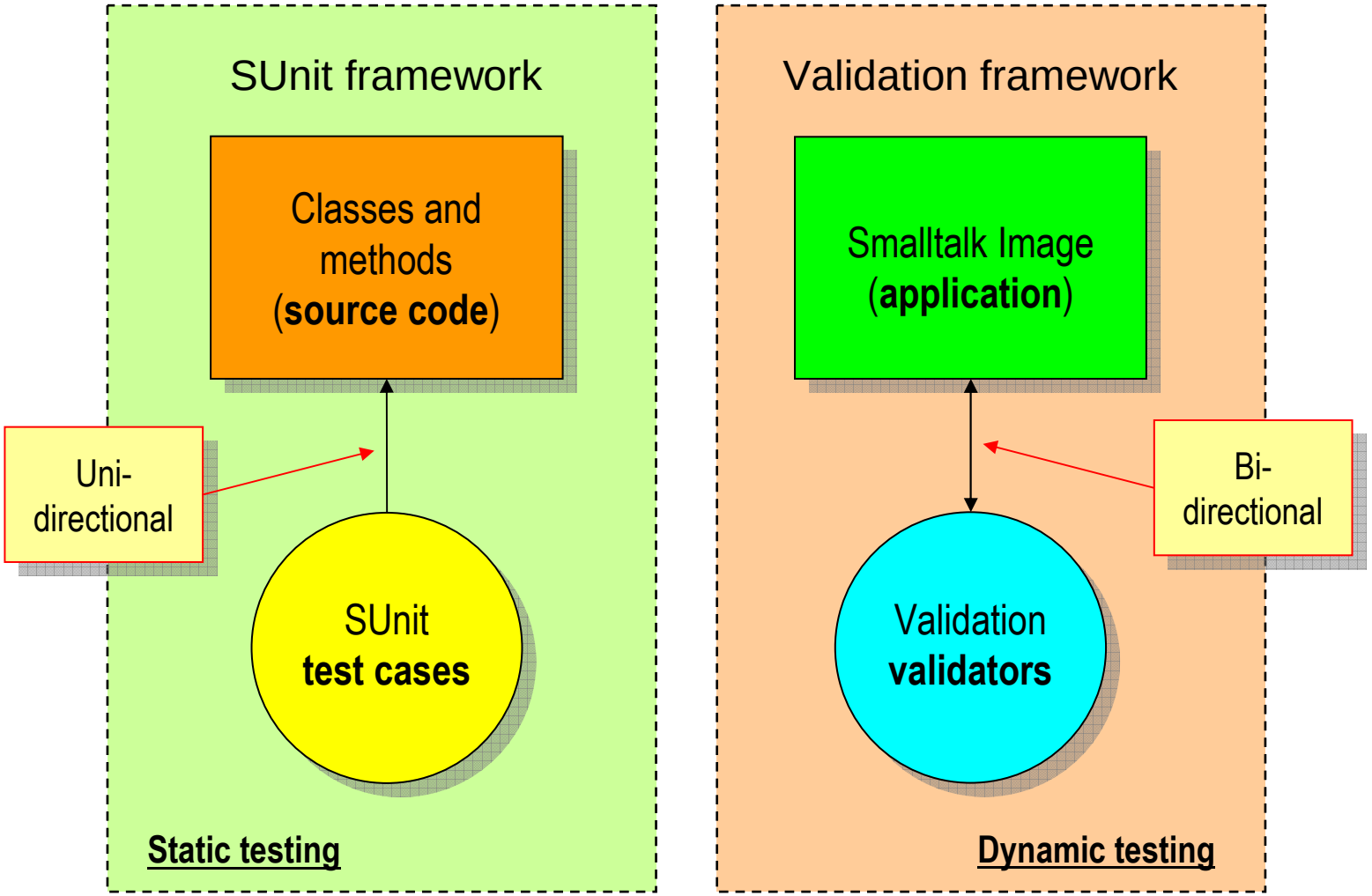
Writing Validatiors




```
namesDoNotCollide
    | names |
    names := self value collect: [:each | each name].
    names size = names asSet size
        ifFalse: [self failBecause: aspect, ' have colliding names']
```

```
includeValidationOf: value
    | result |
    self isDefined: value.
    result := value validate.
    result allValidationFailures do: [:v | v resignal]
```

Comparing territories

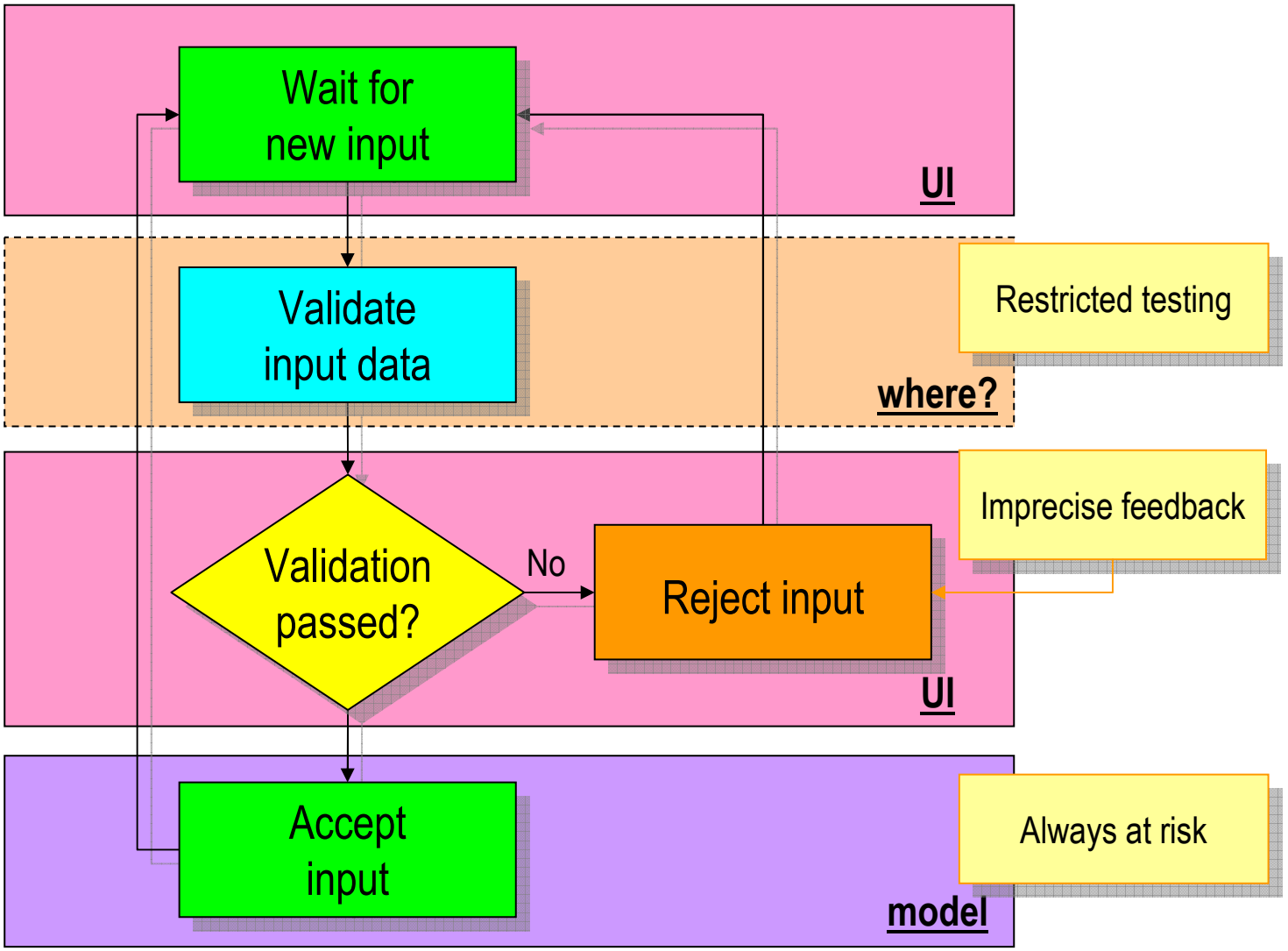


Comparison with Test Cases

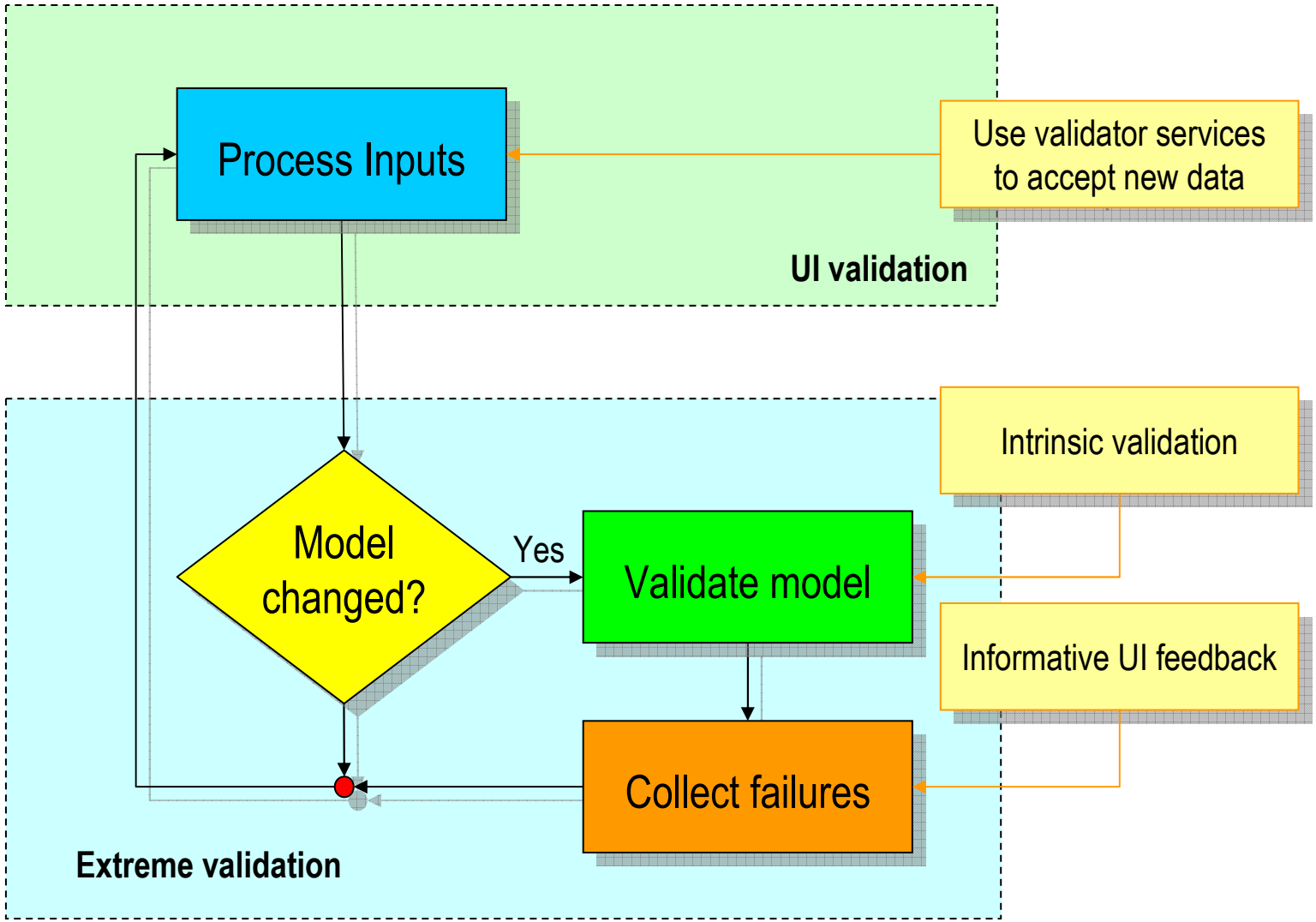
	Test Cases	**Validators**
Purpose	Code correctness	Objects' health
Running Space	Development	Runtime
Objects Involved	Artificial (examples)	Actual (application)
Activated by	Programmers	Application requests
Advice	Programmers	End users and programmers

	Test Cases	**Validators**
Development cost	High – skills required	Negligible – straightforward
Reuse	Low	High because: a) You grow a library b) You chain validations
Running them all takes	Minutes – Hours	Seconds

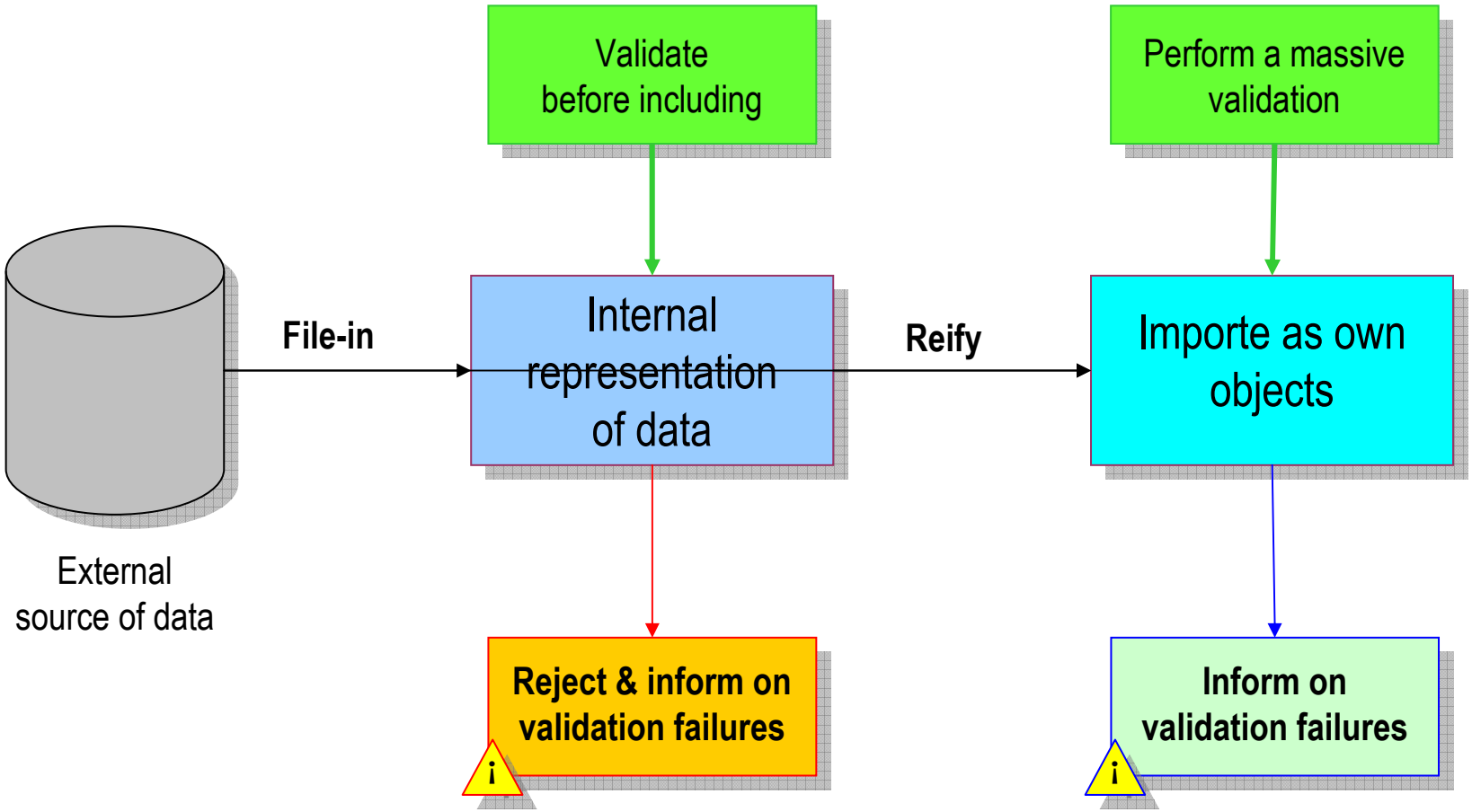
The Conventional Approach



Extreme Validation



Transactional & Informative File-in



When is Extreme Validation useful?

Apply extreme validation to	**So that you**
User inputs	a) Relax strict sequences of steps in the GUI and still ensure the model's health b) Avoid most of the warning dialogs!
Imperfect code	Detect all the side effects of bugs everywhere in the model as early as possible
Transactional applications	Make sure that the model remains valid after every transaction
File-in	Deal with data loaded from external sources (e.g., source code file-in, software upgrades, data importing, XML, etc.)
Structural changes	Make sure the new structure is sound (e.g. merging complex structures)
Test cases	Simplify your tests by asserting that the code does not brake testing objects
Source code	Verify that good practice patterns are honored (see the full paper)
You application	Crearly express all intrinsic rules that are inherent to your objects regardless their presentation on the UI