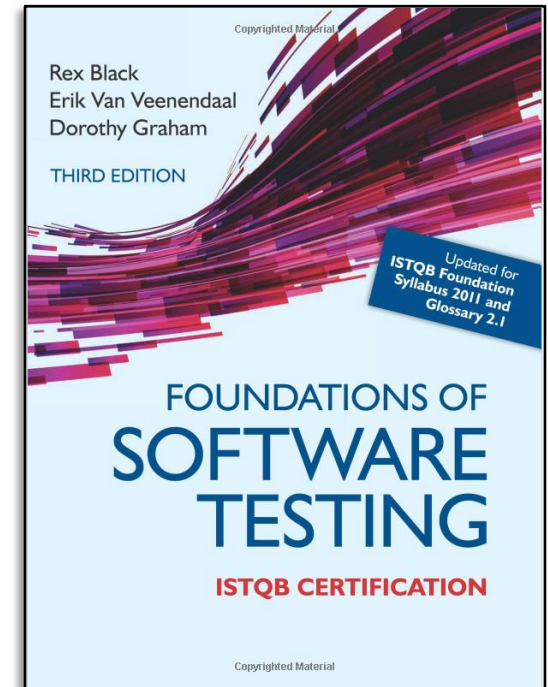
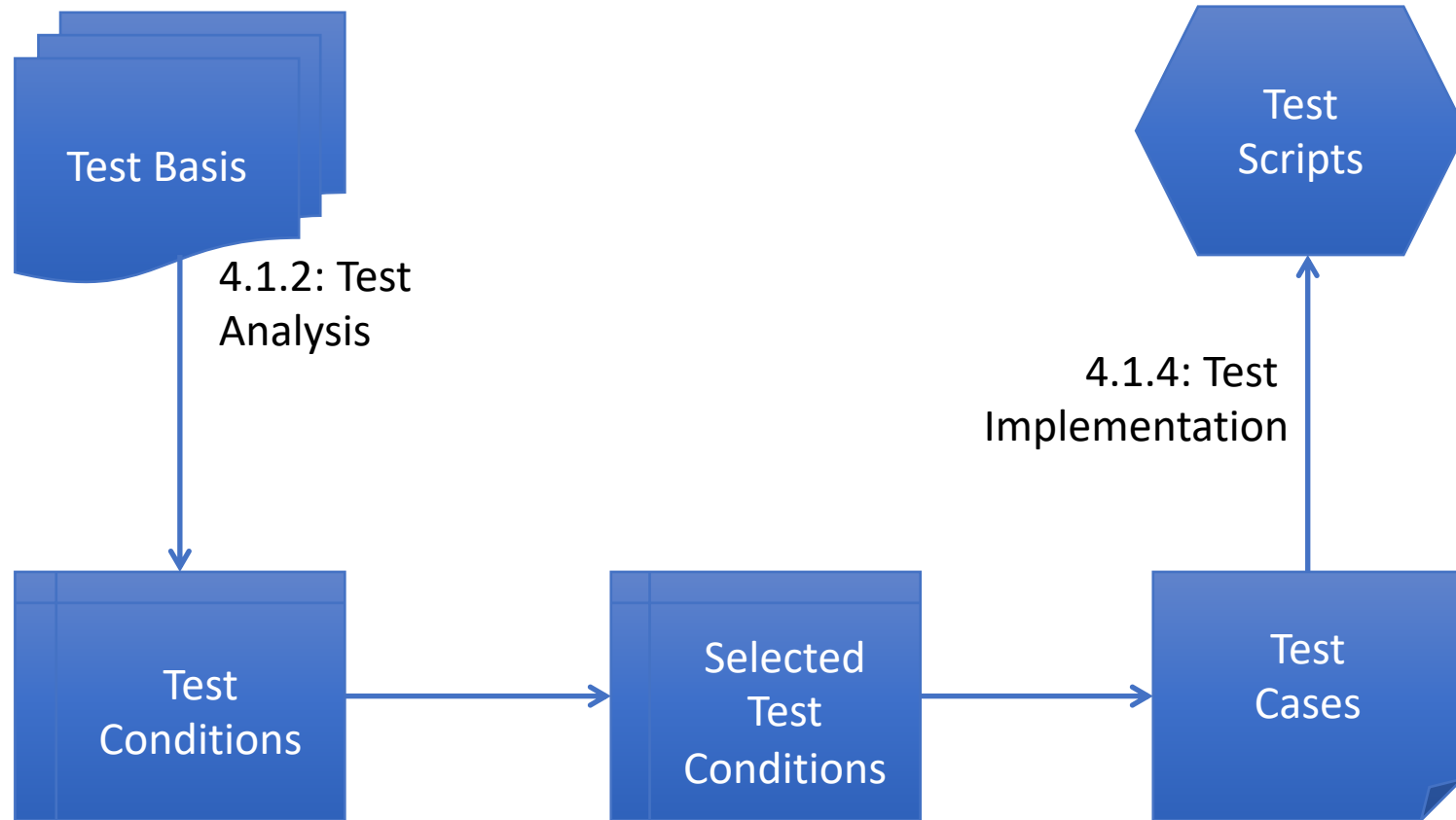


# Model-Based Testing (ISTQB Chapter 4)

Arie van Deursen



# ISTQB Test Design



4.1.3: Test Design

**Test basis:** All documents from which the requirements of a component or system can be inferred.  
Documentation on which the test cases are based

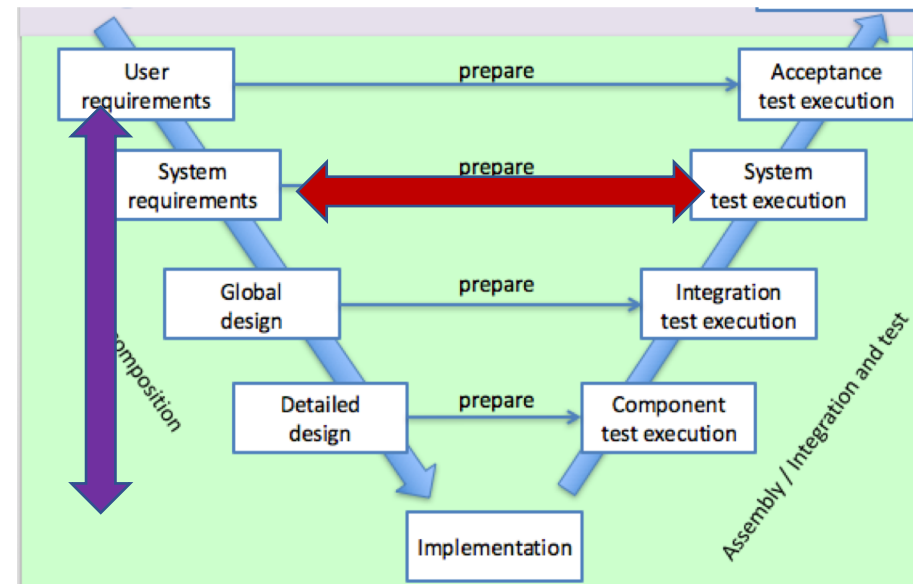
**Test condition:** *An item or event of a component or system that could be verified by one or more test cases, e.g. a function, transaction, feature, quality attribute, or structural element.*

**Test case:** A set of input values, execution preconditions, expected results and execution postconditions, developed for a particular objective or test condition, such as to exercise a particular program path or to verify compliance with a specific requirement.  
[After IEEE 610]

# Traceability

- Link test conditions back to test basis
- Where is a given requirement tested?
- Which requirements does this test case address?

- Horizontal:
  - Within one test level
- Vertical:
  - Between requirement and implementation





# Traceability Management in Practice

## Agile (modern)

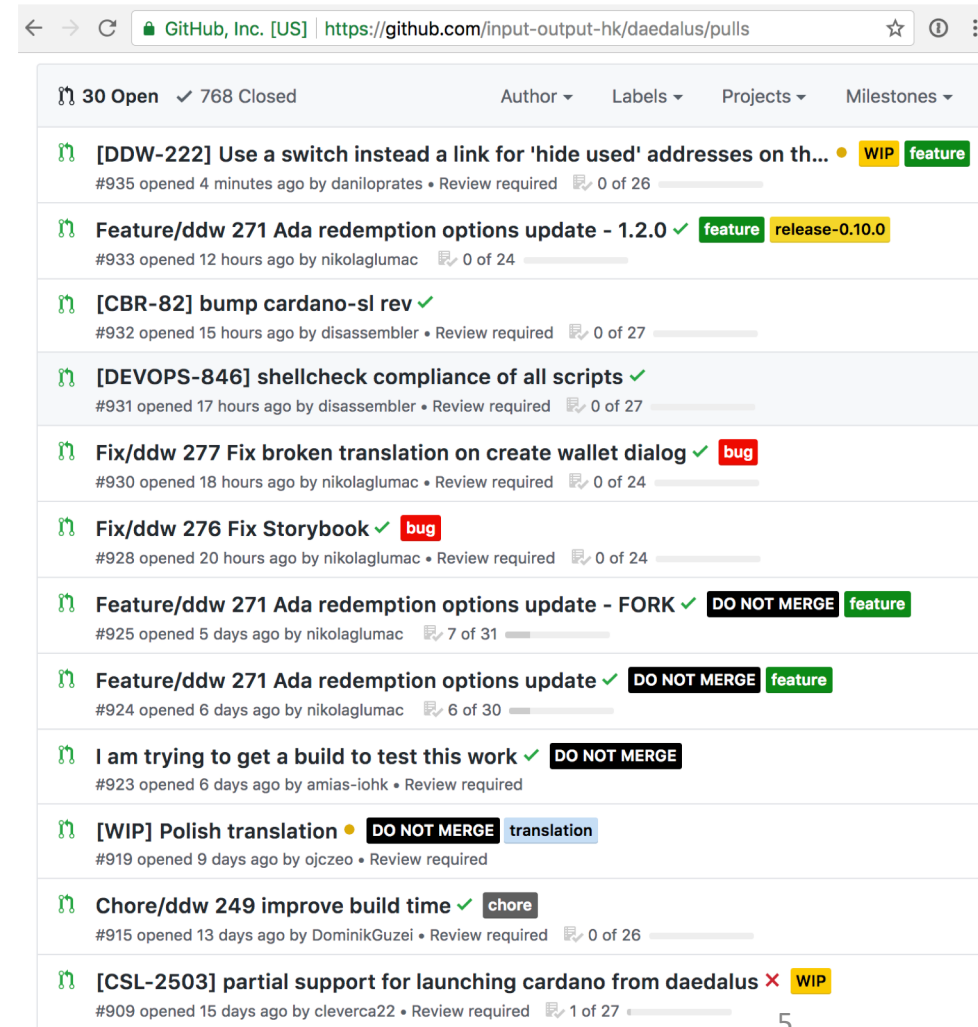
- User stories have issue ID (e.g., GitHub, Jira)
- Pull requests / commits trace back to issues
- Story boards (GitHub, Trello)

## High ceremony (traditional)

- Requirements management system

## Low tech (high maintenance)

- A spreadsheet



The screenshot shows a GitHub pull request list for the repository 'input-output-hk/daedalus/pulls'. The browser address bar shows the URL 'https://github.com/input-output-hk/daedalus/pulls'. The page header indicates '30 Open' and '768 Closed' pull requests. The list includes several pull requests with their titles, authors, and status labels:

- [DDW-222] Use a switch instead a link for 'hide used' addresses on th...** by daniloprates, WIP, feature, #935 opened 4 minutes ago.
- Feature/ddw 271 Ada redemption options update - 1.2.0** by nikolaglumac, feature, release-0.10.0, #933 opened 12 hours ago.
- [CBR-82] bump cardano-sl rev** by disassembler, #932 opened 15 hours ago.
- [DEVOPS-846] shellcheck compliance of all scripts** by disassembler, #931 opened 17 hours ago.
- Fix/ddw 277 Fix broken translation on create wallet dialog** by nikolaglumac, bug, #930 opened 18 hours ago.
- Fix/ddw 276 Fix Storybook** by nikolaglumac, bug, #928 opened 20 hours ago.
- Feature/ddw 271 Ada redemption options update - FORK** by nikolaglumac, DO NOT MERGE, feature, #925 opened 5 days ago.
- Feature/ddw 271 Ada redemption options update** by nikolaglumac, DO NOT MERGE, feature, #924 opened 6 days ago.
- I am trying to get a build to test this work** by amias-iohk, DO NOT MERGE, #923 opened 6 days ago.
- [WIP] Polish translation** by ojczco, DO NOT MERGE, translation, #919 opened 9 days ago.
- Chore/ddw 249 improve build time** by DominikGuzei, chore, #915 opened 13 days ago.
- [CSL-2503] partial support for launching cardano from daedalus** by cleverca22, WIP, #909 opened 15 days ago.



# “If you did not document it, you did not do it!”

- Inspections by Government & Notified Bodies:
  - If you do not follow regulation & your internal procedures (QMS), you cannot guarantee safety & effectiveness.
- Consequences:
  - Delivery stop for sites outside USA and/or close down for sites in the USA.
  - In case of safety (patient) issues & not sticking to the law: Jail for upper mgt.
- At Philips:
  - Inspection Back-office to answer questions fast & accurately



# Test Design Includes Oracle

“Software that applies a pass/fail criterion to a program execution is called a *(test) oracle*”.

## Approaches

1. Comparison against predicted output
2. Self checks (“Partial Oracle” / “Reasonableness check”)
3. Version comparisons



Testing a Sudoku generator/solver?

You can't predict full output

But you can check validity of any solution (9 unique digits on every row, column, square)

6	7	1				2	4	9
8			7		2			1
2				6				3
	5		6		3		2	
		8				7		
	1		8		4		6	
9				1				6
1			5		9			7
5	8	7				9	1	2

# Dynamic Test Design Techniques

- **Specification Based** (black box, 4.3)
  - Equivalence partitioning, boundary value analysis
  - Decision tables, state transition (**model-based**)
  - Use case testing
- **Structure Based** (white box, 4.4)
  - Statement, decision, condition, multiple condition
- **Experience Based** (4.5)
  - Error guessing, exploratory testing

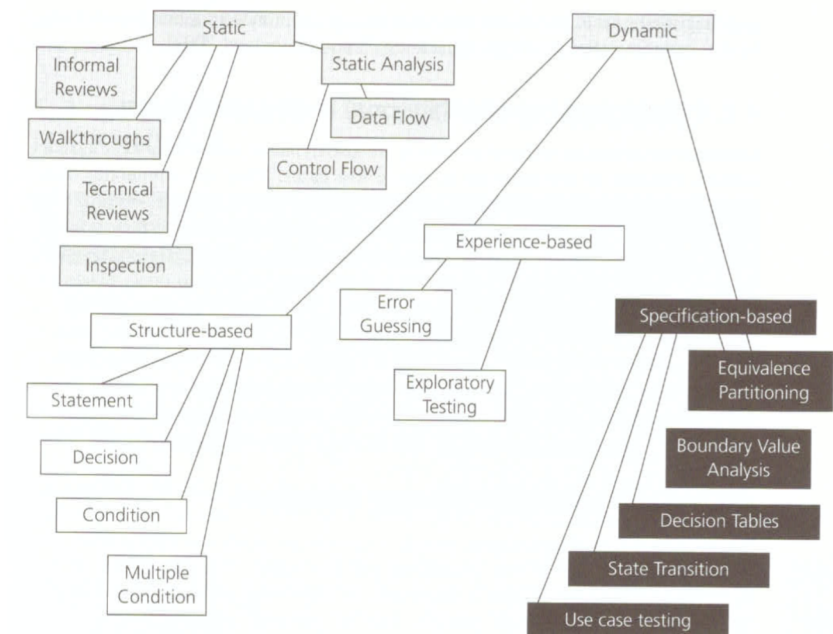


FIGURE 4.1 Testing techniques

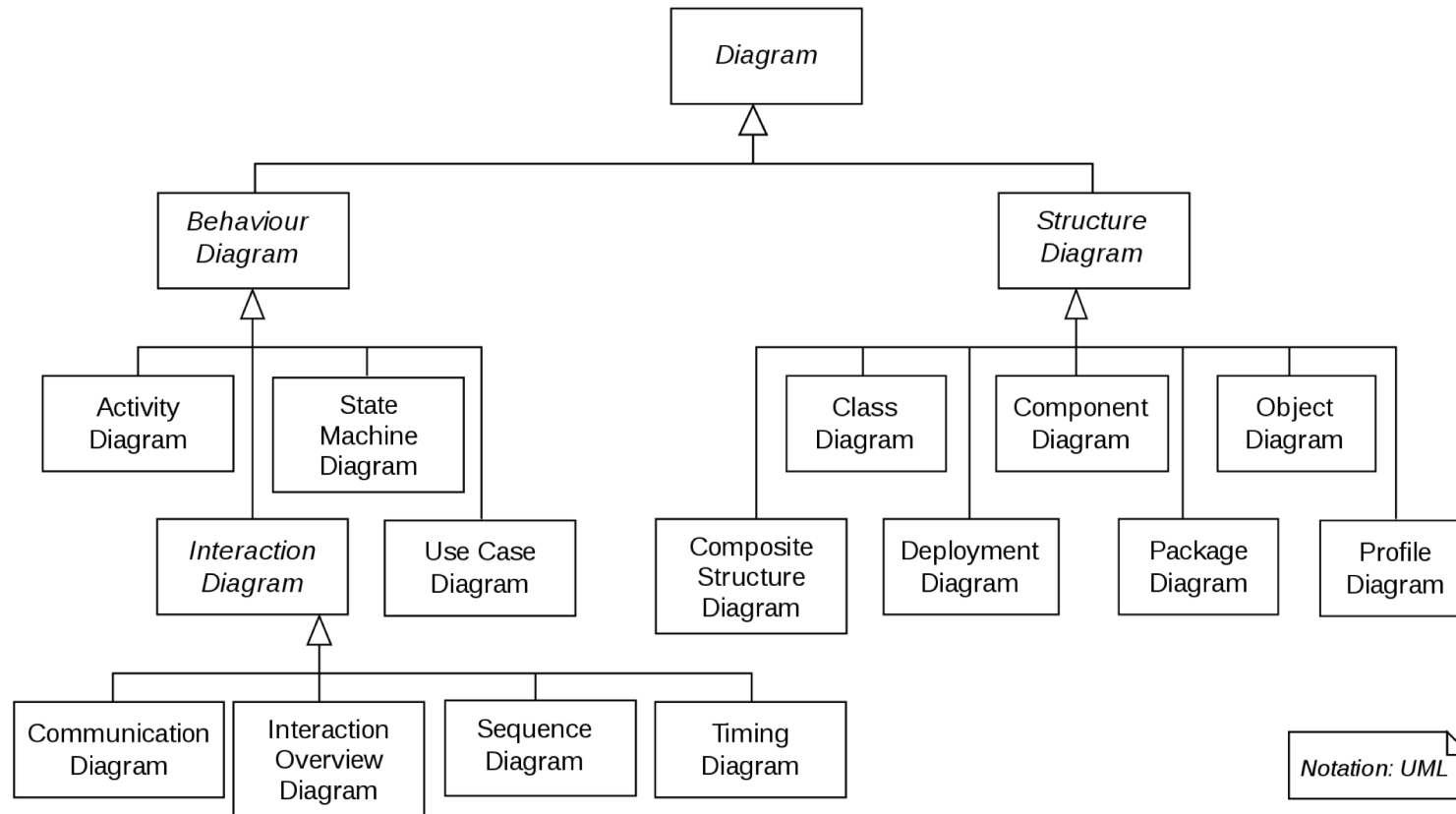


## Model:

- Simpler than artifact
- Preserves (approximates) certain key attributes
- Supports *analysis*



# Model Types in the UML



# Models for Testing

- Models from requirements
  - Meaningful to domain expert
  - Use to obtain test cases that systematically exercise required behavior
- Models from code
  - Meaningful to developer
  - Use to obtain test cases that systematically exercise implemented behavior



# Choices!

- New or old?
- 12 or 24?
- KPN at home?
- Budget?
- Basis?
- No worries?
- Also internet provider?
- ...

KPN Zorgeloos 1 jaar sim-only e X

https://www.mobiel.nl/sim-only/kpn/zorgeloos-compleet-sim-only-1-jaar/481-7

## KPN sim only

Stel zelf je KPN sim only-abonnement samen

[Selecteer je KPN sim only](#) [Informatie over KPN](#)

**Contracttype**

Nieuw  Verlenging

**Contractduur**

12 mnd  24 mnd

Ja  Ik heb **KPN thuis** en profiteer daarom van extra voordelen

**Abonnementsvorm**

Budget  Basis  **Zorgeloos**

Je hebt internet voor thuis van KPN of XS4ALL en profiteert daardoor van €5,- korting en andere voordelen. Hoe meer je hebt, hoe meer je krijgt! In onderstaand aanbod zit de €5,- korting op je abonnementskosten en de dubbele data verwerkt.

De huidige netwerkacties zijn geldig tot en met 03-06-2018.

Kies je bundel

### Je bestelling

<b>Sim only</b>	
KPN Zorgeloos 1 jaar	€ 25,00
Standaard	€ 25,00
KPN Thuis klant	€ 0,00
<b>Totaal per maand</b>	<b>€ 25,00</b>

**Bestel nu**

Excl. verzendkosten | Incl. 21% btw  
Aansluitkosten € 25,00

### Eenvoudig bestellen

abc → →

Vul je gegevens in    Onderteken je contract via iDEAL    Ontvang je simkaart

# A Simple Decision Table

**Decision Table:** models how combinations of conditions lead to given actions (or outputs)

		<i>Variants</i>			
<i>Conditions</i>	International?	F	F	T	T
	Auto-renewal?	T	F	T	F
<i>Action</i>	<b>Price/month</b>	<b>10</b>	<b>15</b>	<b>30</b>	<b>32</b>

International?	F	F	F	T	T	T
Auto-renewal?	T	dc	F	T	dc	F
Loyal?	dc	T	F	dc	F	F
<b>Price/month</b>	<b>10</b>	<b>10</b>	<b>15</b>	<b>30</b>	<b>30</b>	<b>32</b>

With “don’t care” values

International?	F	F	F	F	F	T	T	T	T	T
Auto-renewal?	T	T	T	F	F	T	T	T	F	F
Loyal?	T	F	T	T	F	T	F	T	T	F
<b>Price/month</b>	<b>10</b>	<b>10</b>	<b>10</b>	<b>10</b>	<b>15</b>	<b>30</b>	<b>30</b>	<b>30</b>	<b>30</b>	<b>32</b>

With “don’t care” values expanded

International?	F	F	F	F	T	T	T	T
Auto-renewal?	T	T	F	F	T	T	F	F
Loyal?	T	F	T	F	T	F	T	F
<b>Price/month</b>	<b>10</b>	<b>10</b>	<b>10</b>	<b>15</b>	<b>30</b>	<b>30</b>	<b>30</b>	<b>32</b>

With duplicate variants removed

# Larger Decision Tables

- Decision tables can have many conditions
- In general: N conditions:  $2^N$  variants
- Omitted / non-specified variants?
  - Indicate what “default” behavior is.

# Five Decision Table Test Strategies

All explicit variants: 6

All possible variants:  $2^3 = 8$   
(= all combinations)

International?	F	F	F	T	T	T
Auto-renewal?	T	dc	F	T	dc	F
Loyal?	dc	T	F	dc	F	F
<b>Price/month</b>	<b>10</b>	<b>10</b>	<b>15</b>	<b>30</b>	<b>30</b>	<b>32</b>

Each condition T/F:  
2 cases (TTT, FFF)

All decisions /  
every unique outcome: 4

Each condition AND all decisions = (M)C/DC 17

# MC/DC:

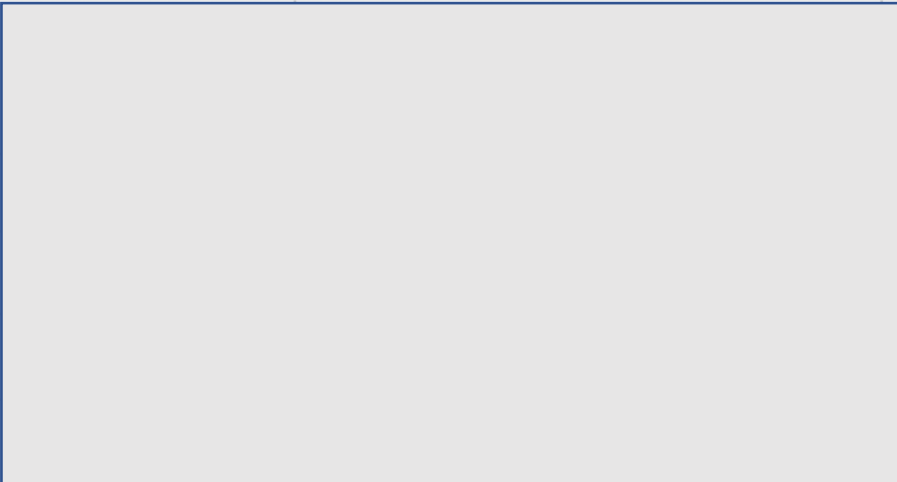
## Modified Condition / Decision Coverage

- **Conditions:** Each condition should be once true, once false
- **Decisions:** Each action should be taken at least once
- **Modified:** Each condition should individually determine the outcome
- For each condition require two test cases that only differ in outcome and that condition

# Finding an “MC/DC Cover”

- Expand decision table
- Pick variants with unique outcome
- Combine with others so that they differ in one condition only

	v1	v2	v3	v4	v5	v6	v7	v8
International?	F	F	F	F	T	T	T	T
Auto-renewal?	T	T	F	F	T	T	F	F
Loyal?	T	F	T	F	T	F	T	F
<b>Price/month</b>	<b>10</b>	<b>10</b>	<b>10</b>	<b>15</b>	<b>30</b>	<b>30</b>	<b>30</b>	<b>32</b>



# Finding an “MC/DC Cover”

- Expand decision table
- Pick variants with unique outcome
- Combine with others so that they differ in one condition only

	v1	v2	v3	v4	v5	v6	v7	v8	mc/dc	
International?	F	F	F	F	T	T	T	T	v4,v8	outcomes 15 and 32
Auto-renewal?	T	T	F	F	T	T	F	F		
Loyal?	T	F	T	F	T	F	T	F		
<b>Price/month</b>	<b>10</b>	<b>10</b>	<b>10</b>	<b>15</b>	<b>30</b>	<b>30</b>	<b>30</b>	<b>32</b>		



# Finding an “MC/DC Cover”

- Expand decision table
- Pick variants with unique outcome
- Combine with others so that they differ in one condition only

	v1	v2	v3	v4	v5	v6	v7	v8	mc/dc	
International?	F	F	F	F	T	T	T	T	v4,v8	outcomes 15 and 32
Auto-renewal?	T	T	F	F	T	T	F	F	v4,v2	outcome 10
Loyal?	T	F	T	F	T	F	T	F		
<b>Price/month</b>	<b>10</b>	<b>10</b>	<b>10</b>	<b>15</b>	<b>30</b>	<b>30</b>	<b>30</b>	<b>32</b>		

# Finding an “MC/DC Cover”

- Expand decision table
- Pick variants with unique outcome
- Combine with others so that they differ in one condition only

	<i>v1</i>	<i>v2</i>	<i>v3</i>	<i>v4</i>	<i>v5</i>	<i>v6</i>	<i>v7</i>	<i>v8</i>	<i>mc/dc</i>	
International?	F	F	F	F	T	T	T	T	<i>v4,v8</i>	outcomes 15 and 32
Auto-renewal?	T	T	F	F	T	T	F	F	<i>v4,v2</i>	outcome 10
Loyal?	T	F	T	F	T	F	T	F	<i>v8,v7</i>	outcome 30
<b>Price/month</b>	<b>10</b>	<b>10</b>	<b>10</b>	<b>15</b>	<b>30</b>	<b>30</b>	<b>30</b>	<b>32</b>		

# MC/DC: N+1 Test Cases

- For a table with N conditions and yes/no actions, N+1 test cases suffice to obtain an MC/DC cover
- Condition C1: Test cases T1 and T1'
- Condition C2: Try to “reuse” earlier test cases T1 or T1'

# JUnit Test Methods from Decision Tables

@Test

```
void internationalExpensive() {  
    PhonePlan plan = new PhonePlan();  
  
    plan.setInternational(true);  
    plan.setAutoRenewal(false);  
    plan.setLoyal(false);  
  
    assertThat(plan.pricePerMonth())  
        .isEqualTo(32);  
}
```

# Junit Parameterized Tests

```
@ParameterizedTest
@CsvSource({
    "true, false, false, 32",
    "true, false, true, 30",
    "false, false, false, 15",
    "false, true, false, 10"
})
void testPlan(boolean inter,
              boolean renew,
              boolean loyal,
              int expected) {
    PhonePlan plan = new PhonePlan();

    plan.setInternational(inter);
    plan.setAutoRenewal(renew);
    plan.setLoyal(loyal);

    assertThat(plan.pricePerMonth())
        .isEqualTo(expected);
}
```

# Cucumber Scenario

Scenario Outline: Determine plan price

Given I need a phone subscription

When I pick international calls to be <international>

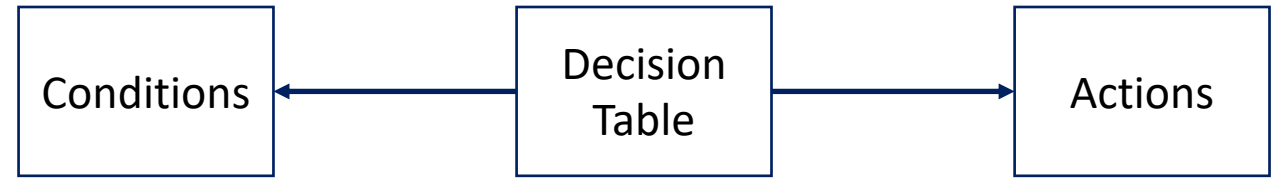
And I pick auto-renewal to be <autorenewal>

Then the price per month is <price>

Examples:

international	autorenewal	price
true	true	30
true	false	32
false	true	10
false	false	15

# Controllability and Observability



- Can conditions be easily set?
  - Environmental conditions, exceptions, ...
- Can actions be easily observed?
  - Side effects, state changes, ...
- With mocking
  - Mock condition classes to set inputs
  - Mock action classes to observe effects
- Decision table test cases focus on *combinations of conditions!*

# Non-binary “decisions”

- Decision tables can be generalized to non-Boolean conditions
- Most testing strategies remain possible
- To manage combinatorial explosion dedicated combinatorial testing techniques may be more suitable (e.g. “pairwise testing”).

International?	F	F	F	F	T	T
Auto-renewal?	T	T	F	F	F	T
Data limit	0	8	0	8	∞	∞
<b>Price/month</b>	<b>10</b>	<b>12</b>	<b>15</b>	<b>17</b>	<b>30</b>	<b>32</b>

Three possible values:  
0Gb, 8Gb, or no limit.

Table has  $2 \times 2 \times 3 = 12$  choices  
Not listed => impossible



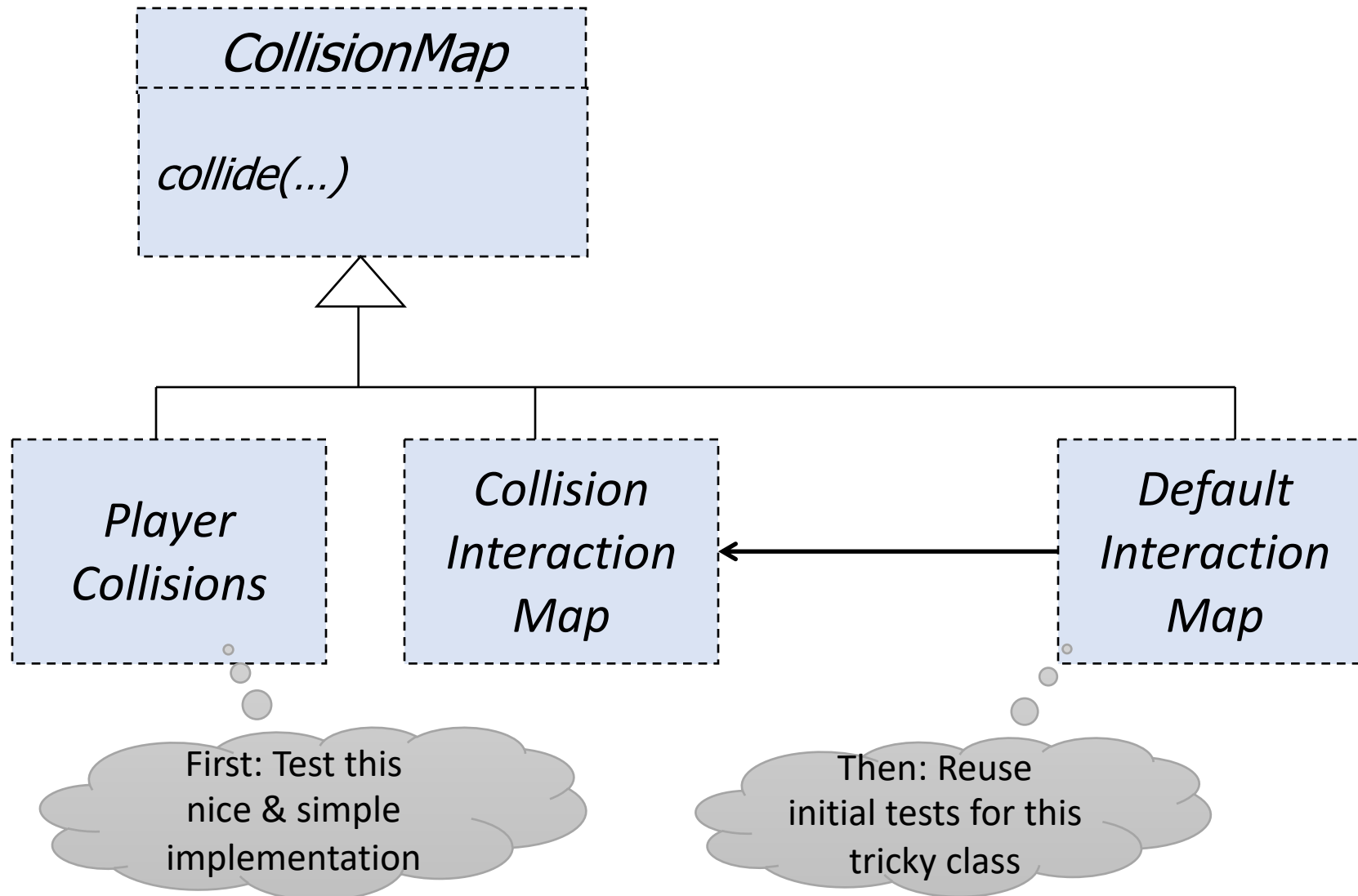
# Design Guidelines

1. Keep conditions *independent*
2. Use DC values to reduce number of variants
3. Avoid overlap between DC values
4. Try to add default column
5. If conditions are mutually exclusive consider using non-binary logic
6. If most conditions are non-binary consider combinatorial testing

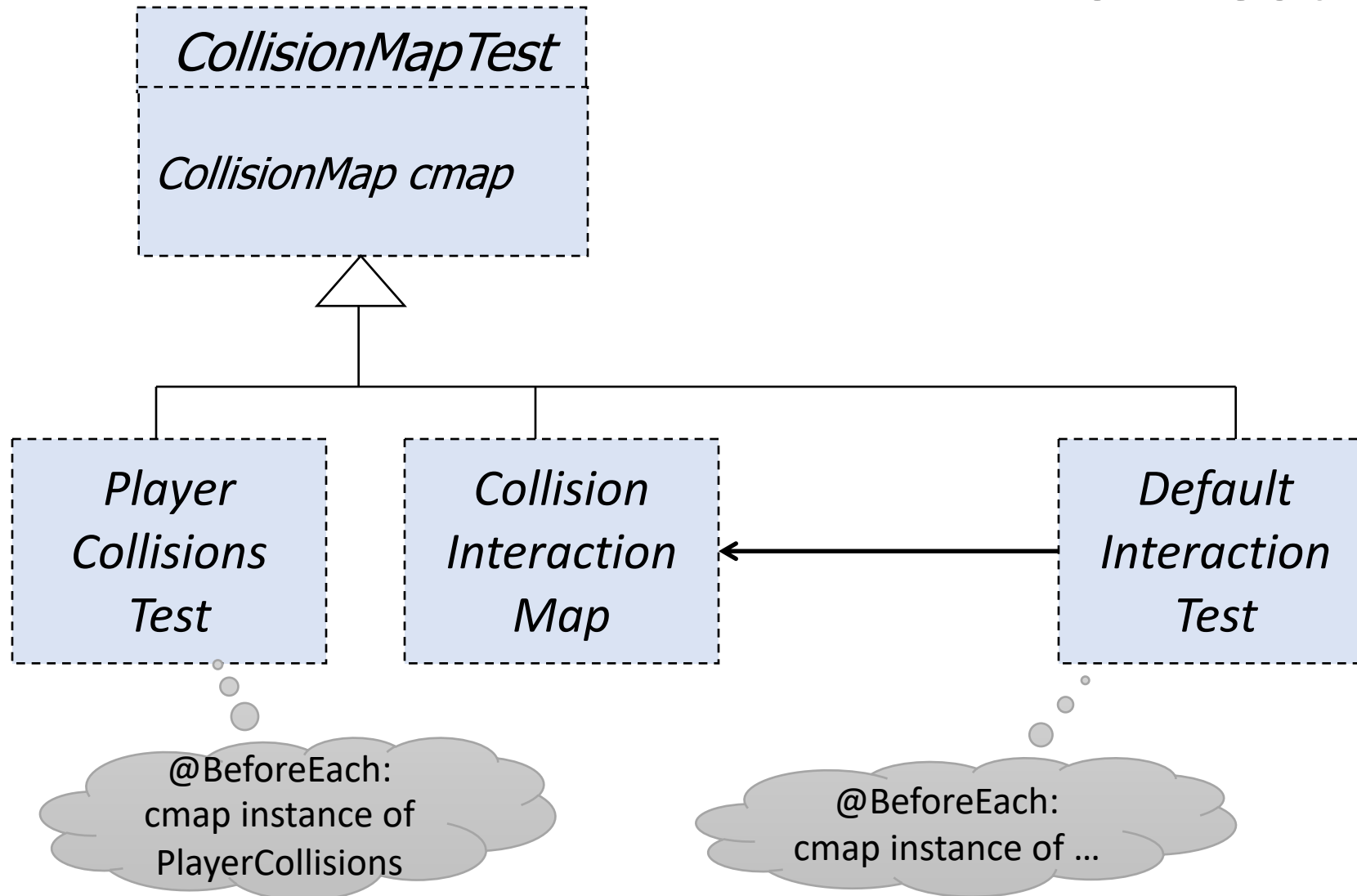
# JPacman Collision Decision Table?

- Conditions: collider / collidee type (classes)
- Rules: Collider / collidee combination
- Action: Die, eat, ...
  
- Two alternatives:
  - Binary table (with disjoint conditions)
  - Non-binary table (simpler)

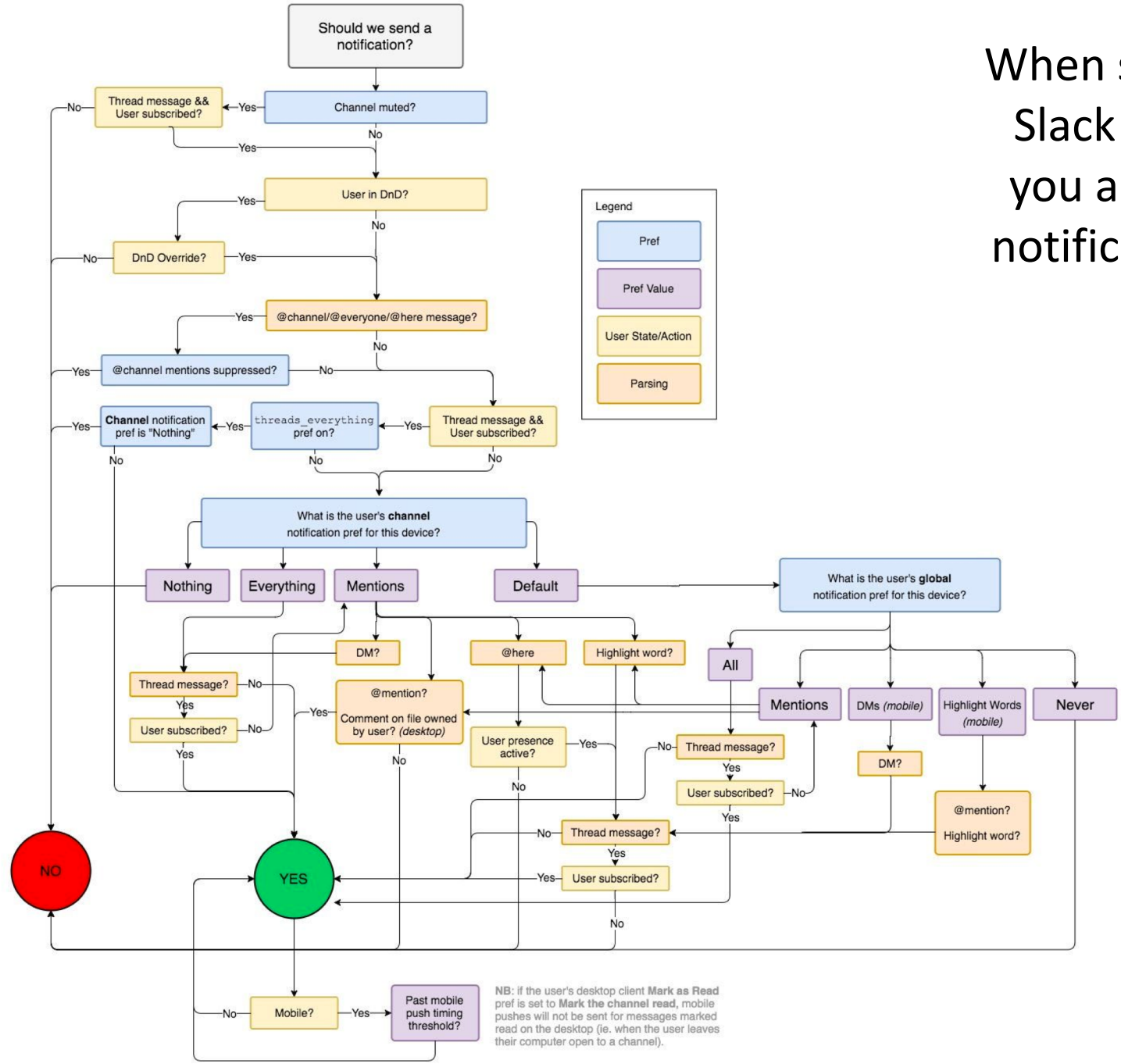
# The Collision Hierarchy



# Collisions: “Parallel Class Hierarchy” for Testing

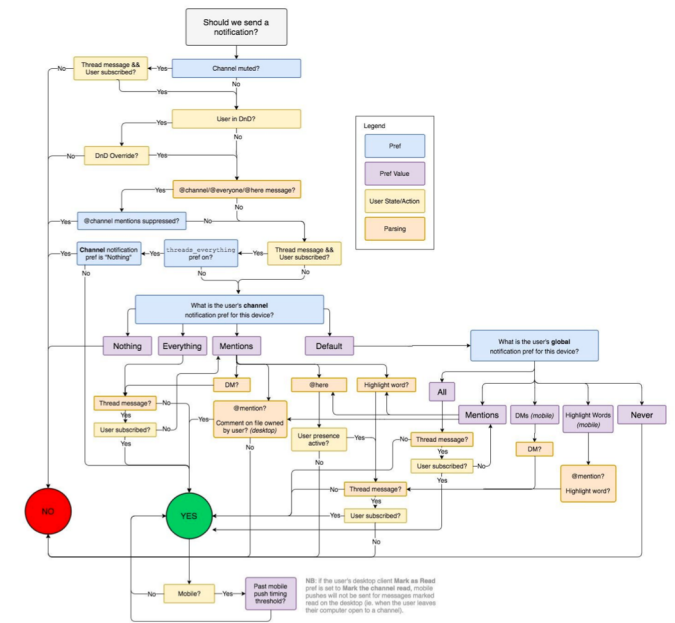


# When should Slack send you a push notification?



# Rethinking Slack Notifications

- How many independent conditions?
- Use non-binary decisions for, e.g., notification preferences (nothing, everything, mentions, default)
- Create pairs of columns in which changing one condition affects the outcome. If possible.
- Substantial duplication in full table. Richer logic beneficial.



Should we send?	No	No	No	No	No	No	Yes	No	Yes
Channel muted	T	T	T	T	T	T	T	T	T
Thread message	F	F	T	T	T	T	T	T	T
User subscribed	F	T	F	T	T	T	T	T	T
User in DnD	-	-	-	T	T	T	T	T	T
DnD override	-	-	-	F	T	T	T	T	T
channel/here/everone	-	-	-	-	T	T	T	T	T
atchannel suppr	-	-	-	-	T	F	F	T	T
threads_ever thread on	-	-	-	-	-	T	T	F	F
channel preference	-	-	-	-	-	"Nothing"	!"Nothing"	"Nothing"	"Everything"
mention status	-	-	-	-	-	-	-	-	-
mention on own file	-	-	-	-	-	-	-	-	-
User presence active	-	-	-	-	-	-	-	-	-
global preference	-	-	-	-	-	-	-	-	-
mobile	-	-	-	-	-	-	T	-	T
past-push timing threshold	-	-	-	-	-	-	F	-	F

Should we send?	Yes	No	Yes	No	Yes	Yes	Yes	No	No
Channel muted	T	T	T	T	T	T	T	T	F
Thread message	T	T	T	T	T	T	T	T	-
User subscribed	T	T	T	T	T	T	T	T	-
User in DnD	T	T	T	T	T	T	T	T	T
DnD override	T	T	T	T	T	T	T	T	F
channel/here/everone	T	T	T	T	T	T	T	T	-
atchannel suppr	T	T	T	T	T	T	T	T	-
threads_ever thread on	F	F	F	F	F	F	F	F	-
channel preference	"Default"	"Default"	"Default"	"Default"	"Default"	"Default"	"Default"	"Default"	-
mention status	at-mention	at-mention	here	here	highlight	DM	hilight	hilight	-
mention on own file	T	F	-	-	-	-	-	-	-
User presence active	-	-	T	F	T	-	-	-	-
global preference	Mentions	Mentions	Mentions	Mentions	Mentions	DM	Hilight	Never	-
mobile	T	-	T	-	T	T	T	-	-
past-push timing threshold	F	-	F	-	F	F	F	-	-



Should we send?	No	Yes	No	Yes	No
Channel muted	T	F	F	F	F
Thread message	F	F	F	F	F
User subscribed	T	T	T	T	T
User in DnD	F	F	T	T	T
DnD override	F	F	F	T	T
channel/here/everone	F	F	F	F	T
atchannel suppr	T	T	T	T	T
threads_ever thread on	T	T	T	T	T
channel preference	Everything	Everything	Everything	Everything	Everything
mention status	-	-	-	-	-
mention on own file	-	-	-	-	-
User presence active	-	-	-	-	-
global preference	-	-	-	-	-
mobile	T	T	T	T	T
past-push timing threshold	F	F	F	F	F

# Decision Tables

- Concise model of complex decision logic
- Increase understanding of
  - The application domain
  - Your code base
- Cover essential logic in manageable test suite using MC/DC strategy