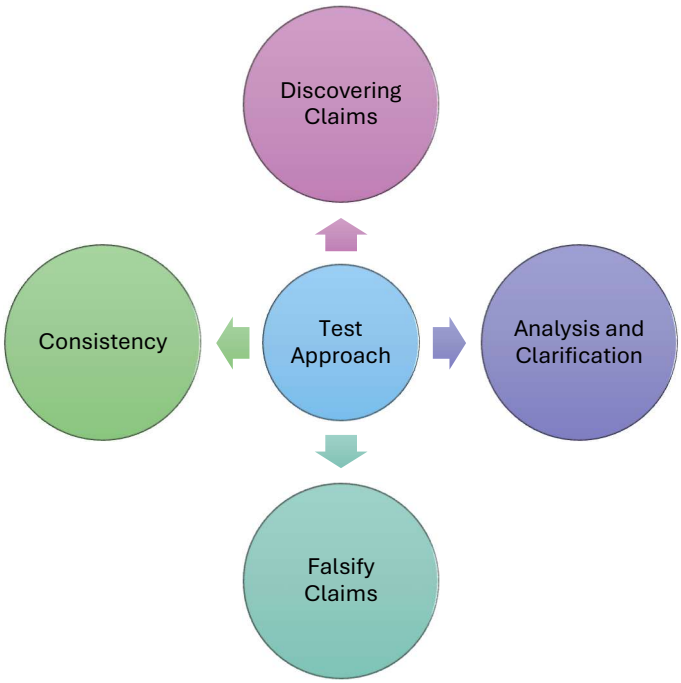
Heuristics for Claims-Based Testing



Test Approach: Different ways we can test claims

Discovering Claims: Ways to find claims we can test or study

Analysis and Clarification: Questions to ask or approaches to studying claims

Falsify Claims: Ways to demonstrate a given claim might not be true

Consistency: Ways of checking whether claims are consistent with each other, and business need.

Test Approach

We can test claims almost any way we want. Below is a suggested, broad, simplistic approach that starts with idea collection, analysis, and their choices for ways to test. Adjust as you see fit or make up yourself.

1. **Make a list of the claims**

   See below for discovering claims. Once you discover them, make a list of the claims you want to investigate or test. The list may be simple and short, or detailed and long. Adjust as appropriate to your needs.

2. Build falsification information

For every claim, note what would be necessary to make that claim true. For example, if a buyers club application claim states it rewards users based on whether they have more than 1000 club points, the claim would require there be a way to check club points, that club points even exist and are assignable to users, and that users ought to be able to achieve more than 1000 points. Any of those found false in analysis or testing would falsify the claim. This list is useful during testing.

3. Build models to describe claims

Construct a state-based model that encompasses one or more of the claims. Are the facts made in the claim consistent with the rest of the model? Do the conditions necessary for the claim to be true, or the outputs of acting on the claim lead to problematic states? Perform tests that exercise traversals of the model that cover the claims in different ways.

Break claims into the individual behaviors, features, and stories that satisfy the claim and test them on their own.

4. Construct and test scenarios to cover claims

For every example scenario in tutorials, introduce variations of that scenario. Use different data, store and retrieve persisted data in different kinds of storage, integrate with similar but different external components, execute a different action than the example shown in the tutorial.

Inject whatever scenario is covered by a claim into as real a use of that scenario as possible. Look for ways the larger scenario or purpose no longer works.

5. Various Claim Testing Ideas

Analyze and test claim failure modes

Perform a failure analysis (see FEFMEA for a short and fast failure analysis technique) which covers parts of the system or product that encompass the claims. What failures are possible at, prior to, and immediately after the parts of the system where the claims are focused? Perform tests that exercise these failure points.

Examine claims for potential against testing under failure modes, stress conditions, or other cases that make satisfying the claim difficult.

Perform data analysis of claim

Perform an analysis of any data relevant to the claim. Are there permutations of the data inputs, transformations, or state which would violate the claim or demonstrate it untrue?

Analyze users affected by claim

Find somebody who is a target end user of the system and observe them using the product in ways that the claim would cover. Does anything you observe demonstrate the claim false?

Examine monitors relative to claim

Create monitors of product telemetry and diagnostic logs that measure any piece of data described in the claim. Use this to establish measurements and analysis of real-world usage if the claims are ever violated.

Discovering Claims

Engineering and Product Sources

Any document we create or maintain for sake of building the product may be the source of claims worth testing and checking. These could include any of the following

- Requirements document
- Development Documents
- Jira Tickets and User Stories
- Design mockups and prototypes
- Source files (ever checked the name of a class or method against what it does? How about an API?)

Legal, Marketing and Sales Sources

Read the marketing materials and make notes of statements that claim to create groundbreaking conditions such as no longer needing certain people, talent, procedures, equipment. Look for comparisons of competing products and note exactly which capability the product claims to match, improve or obsolete. Look at the “make me sick/make me better” rhetoric so common in marketing and sales literature and see if that claim is met by the product.

Examine End User Licensing agreements, contracts or other documents for statements about product behavior. In some cases, these documents may oblige the product team to guarantee certain behavior.

Training Materials

Watch tutorial videos and read tutorial content. Match the tutorial steps and outcome with any claim of making work easier, more productive, less error prone. Follow the steps yourself and see if everything aligns. Are commands and controls the same? Are messages and help text and display identical? Do different events and actions appear, or does the product wind up in a different state?

In-Product Claims

Instructions on screen, error messages, help text, and status messages make claims about what the product will do or what it has done. Are these consistent with actual product behavior? Are they true?

Social Media Sources

Look at the social media posts of the CEO or other evangelists of the product. Pay attention to anecdotes where certain things happened, certain outcomes were gained. Look for percentages and quantities.

Scan community discussion groups and support forums. Observe what is stated there, especially by representatives of the company or users special relationships to the company. Talk with end-users and stakeholders to discover if there are unstated claims that they have come to believe about the product that may not be recorded anywhere.

General Techniques

Look for words and phrases that tip off to the presence of a claim, especially hyperbolic or exaggerated claims. Some of those might be “Replaces…”, “Do away with…”, “Simplifies…”, “Increase productivity X%…”, “You no longer have to….”, “One-stop-shopping for…”, “Every…”, “Faster..”, “Just like a human would…”, “Game changer…”, “Effortlessly…”, “Instant…”, “Unmatched…”, “Accurately…”, “Zero cost…”.

Copy source content into an LLM, preceded by a prompt like the following: “***Examine the following text, and extract a list of claims made in the text about the product "<product name>". Present the claims as a bulleted list of single sentences***.” Review the list for claims that seem suitable for exploration.

Explore feature areas for which you cannot discover claims. Ask stakeholders and product owners about why the gap is there. Explore those features for trouble that may exist in areas ignored.

Prioritize claims based on business, end-user, and stakeholder needs. Establish which of the claims presents the most risk if they are not true, or which present the most risk to implement.

Falsifying Claims

Is the claim credible?

For every claim, perform a credibility analysis. If something is supposed to produce a certain percentage increase in some work product, what volume does math suggest from that increase and is it credible? Look for swapped goals, such as faster typing speed versus time thinking about what the system needs.

Blind obedience

Read the documentation and follow instructions blindly without adjustment. When following instructional steps blindly, make note of every case where you had to take a step that was not in the instructions.

Experimental falsification

Craft a procedure capable of falsifying the claim. If the claim is that productivity gains some percentage, exercise whatever that productivity is and measure impact. If the claim is that some activity is easier requiring expertise, make note of every time you had to step in with the same expertise the claim says is no longer needed.

Self-falsification

Look for claims that contradict each other or claims that describe mutually exclusive conditions that cannot both be true.

Does the claim violate desirability?

Assess the claims for desirability. Are there negative consequences of acting on what the claim implies? Are there risks to the end user, the work environment, or society at large if the claims turn out to be true?

Analysis and Clarification

Is the claim sensible?

Does the claim make sense? Check the claim against any logical fallacies or factual contradictions. Look for triggers words such as “Never” and “Always” and other words that are easily contradicted with single exceptions.

Is the claim supported by data and evidence?

Some claims rely on data and evidence to support them. For example, if a given product claims to make all database queries faster, does the product team have test results showing such an improvement? Does the evidence support the claim as written? In the database queries example, perhaps only certain types of queries are faster and not all.

Do examples show any problems?

Create an example case of each claim. Make the example as detailed and exact as possible. Were there any details you could not complete? Are there any problems that arise when considering the example. Make multiple examples with slight changes to reflect different ways the claim may be expressed.

Do claim relationships show any problems?

Compare product claims against each other. Are there relationships between claims? Are there dependencies between claims? If one claim depends on another for data, behavior, or support, is that capability described in the other claim?

Do claim dependencies present any issues?

Look for dependencies on external systems, actors, data, services, or environments. Make a list of those dependencies and what the claim relies on. Check whether those external dependencies are legitimate, if what the claim relies on is true and real.

Consistency

Do the claims support or contradict each other?

Are claims made in marketing materials (e.g. front page of the website) supported by documentation? Can you find the claim supported in product documentation with instructions on how to use the aspects of the product the claim is about?

Are claims complete or insufficient?

For each claim, construct the features, behaviors and use cases that support the claim. Are there gaps in that list? Do all the features behave as the claim states?

Are claims consistent with user needs?

Examine user feedback for mentioning problems, feature requests or otherwise that support the need or interest in the product claim.

Learn what you can about end user needs and what they do as part of their life or job. Do the claims match what you learned?

Are claims about competition true?

Examine competitive products. Are claims made about competitive products true? Do they offer the same capabilities mentioned, or lack in the way mentioned?

Are claims consistent with the work the team is doing?

Take your list of claims to other members of the product team and ask them if what they are working on is consistent with the claims or not.

Take the claims to stakeholders and ask them if the claims are consistent with their interests, needs and intents for the product.

Are work items consistent with claims?

Are the claims supported by product requirements, do product stories implement what the claims make possible?

Are claims consistent with prior versions?

Check prior versions of the product to see if claims are consistent across versions and releases.