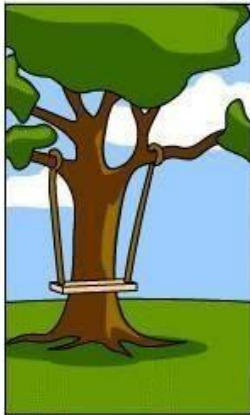


Requirements



What the customer said
that they wanted



How the Sales Rep
understood it



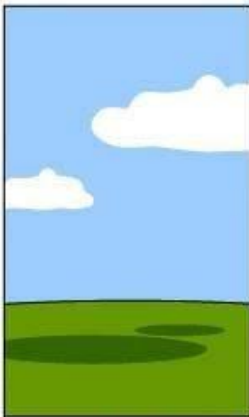
How Solution Mngmnt
wrote the requirements



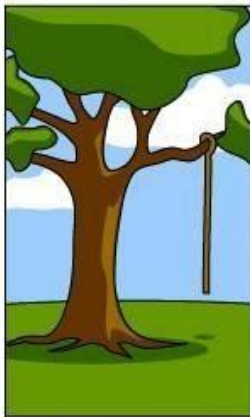
How the Developers
coded it



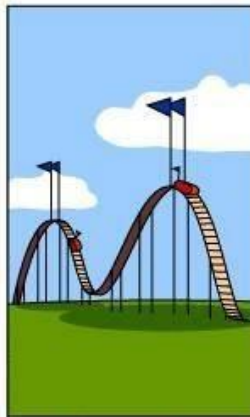
How Marketing
described it



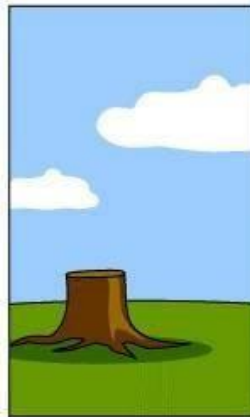
How the project was
documented



What Services
implemented



How the Customer was
billed



How it was supported



What the Customer
really needed

No matter your product or process, always create requirements.

- Used in all development philosophies (Waterfall, Agile, Prototyping, Spiral, eXtreme Programming, etc.).
- Especially valuable in long-lead waterfall-types or client-specific contractual requirements.

And always do it first!



Elicit requirements by connecting with your user

- Be a user yourself (but a double-edged sword!).
- Talk with users informally (hallway chats, mixers)
- Talk with users formally (interviews, surveys, diary studies, field studies, forced ranking)*.
- Build low-fidelity prototypes (mocks, UX prototypes, eng prototypes).
- Launch and get feedback early (“launch and iterate”).

* Results are complex enough to merit an entire specialty: user experience research (UXR).

Write them carefully.

- Describe user-visible attributes.
- Represent collaborators' key goals.
- Leave UX / eng as unconstrained as possible.
- Specify constraints rather than solutions.
- Avoid rigid templates / formats.
- Avoid "and" (break it into two if you want "and").
- Quantify the goal whenever possible.
- Assign priorities.
- Clarify dependencies.

Priorities

- P0 (Must-have / Critical): Absolutely essential, system cannot launch without it.
- P1 (Should-have / Important): High value, but the system can still function without it in the first release.
- P2 (Nice-to-have / Optional): Enhancements or future improvements, not urgent.

Online Food Delivery Application

P0 (Must-have Requirements)

1. The system shall allow users to create an account and log in securely using email/phone and password.
2. The system shall allow users to browse restaurants and menus in their area.
3. The system shall allow users to place an order with at least one payment method (e.g., credit card).
4. The system shall provide real-time order tracking (from restaurant to delivery).
5. The system shall allow delivery drivers to accept and deliver orders.

P1 (Should-have Requirements)

1. The system should support multiple payment methods (credit card, PayPal, digital wallet, cash on delivery).
2. The system should allow users to rate and review restaurants after delivery.
3. The system should provide push notifications for order status updates.
4. The system should allow users to save favorite restaurants and past orders.

P2 (Nice-to-have Requirements)

1. The system may support AI-based recommendations of restaurants and dishes based on user preferences.
2. The system may allow users to schedule orders in advance (e.g., order now, deliver at 7 PM).
3. The system may provide a loyalty/reward points system for frequent users.
4. The system may offer gamification features (e.g., badges for frequent orders, streaks).

Document everything!

- User features
- Performance and System Health
- Reliability
- Scalability
- Warranties or maintenance goals
- Possible or likely future goals
- Target platforms or environments
- Regulatory and legal
- External documentation, user “help”
- Marketing claims
- Logging and success metrics
- Manual testing guides
- Accessibility
- Internationalization, localization, language support
- Troubleshooting guides
- Leak prevention
- Threat models and security guarantees
- User privacy
- Simplicity and usability

Document Everything! (Examples)

- [Features] A short video is captured of ± 2 sec around each shutter press.
- [Performance] The short video will start playing within 200ms of opening the image in 99 percent of cases.
- [System Health] The short video won't exceed 2MB in any case.
- [Reliability] The video loss tolerance is 100 ppm of times the trigger conditions are met.
- [Scalability] The short video will sustain 5 shutter presses per second for 5 seconds on a "clean" device.
- [Warranties, maintenance] The short video feature will not be removed from any device that included it at launch.
- [Possible future goals] The short video may someday share storage space with the primary image using temporal compression formats.
- [Target environments] The short video will run on Pixel 3+.

Document Everything! (Examples)

- [Regulatory] The short video will be subject to only non-biometric algorithmic processing.
- [External documentation] The public Help Center will describe the short video feature.
- [Marketing] The short video feature will be globally available on launch day.
- [Logging and dashboards] Add a field to per-image logs including whether the condition was triggered and whether the short video was saved.
- [Accessibility] The aural description of the on-off toggle will be “short videos will be captured; tap to turn off” when turned “on”.
- [Localization & Internationalization] The feature will be available in all device launch markets (link).
- [Troubleshooting] All triggering conditions are documented for customer support and this document is updated once per release.

Document Everything! (Examples)

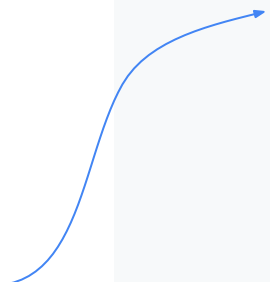
- [Leak prevention] The phrase “short video” will be added to the binary scanner’s database.
- [Security guarantees] The short video will be processed exclusively by the HW and ISP, not SW.
- [Privacy guarantees] The user can turn off capturing short videos.
- [Simplicity] The short video will be captured automatically using the existing shutter button press.

Prioritize ruthlessly

If everything is a “Priority 0” (P0), then nothing is!

- P0 means we’d be embarrassed not to have this.
- P1 is what makes the feature better than the competition.
- P2 is nice to have.

Consider the example of a simple “camera” app.



P0	Takes photos.
P0	Takes videos.
P0	Crashes <0.01% of sessions.
P0	Opens in <1000ms at the P90.
P1	Takes slow motion videos.
P1	Takes time lapse videos.
P1	Does 4K30.
P2	Supports manual photography controls.
P2	Supports RAW capture mode.

Avoid “feature creep”.

What is it?

Requirements are constantly added, which tends to delay release past expectations. That damages morale and may violate contract obligations.

Why does it happen?

Developers like building features. Marketing and sales like to brag. Users always want more. Executives can touch them.

How do I avoid it?

Make timeline a requirement (SMART). Have frequent, regular releases (e.g., monthly). Be realistic about P0s.

Document the requirements you're excluding!

Non-Requirements

- **No legacy devices** because Tripod Mode requires a new HAL architecture for digital zoom that cannot be backported.
- **No slow motion at 4x or 8x speeds** because EIS does not work in slow motion today, and we don't want to meet the 2x - 4x faster per-frame performance budget (relative to 60fps, which is already supported).
- **No photo modes**, such as Camera, Night Sight, Portrait, Photo Sphere, or Panorama, because in these modes you only need proper framing for one instant, vs. in video where you must maintain framing for minutes.
- **No combination of this plus Cinematic Movements**, either (a) by entering Cinematic Movements and then enabling Tripod Mode or (b) enabling Tripod Mode and then moving the camera, because

- Philosophically, Cinematic Movements is about moving the camera and Tripod Mode is the opposite.
- This is effectively a fourth mode of stabilization that requires tuning, evaluation, maintenance, etc., alongside Lock, normal, and Cinematic Movements each alone.
- Tripod Mode must operate without lookahead in order to provide a reasonably-accurate preview, but Cinematic Movements requires significant lookahead to be reasonably smooth.
- **No way to activate Tracking Stabilization from within Tripod Mode**, because
 - The entire purpose of Tripod Mode is to stay still no matter what, and that's the opposite of Tracking Stabilization. They cannot be combined intelligently.
 - If tapping while in Tripod Mode activated Tracking Stabilization, it would effectively prevent the user from manipulating exposure and focus controls without exiting Tripod Mode.
- **No warning for zooming out that are visible to a distracted user**, because we assume the user is looking at the screen while zooming.
- **We do not protect video quality** to warn the user when Tripod Mode is automatically disabled. We prioritize (A) getting the user's attention above (B) protecting the video's quality for that moment, so that the entire video isn't ruined.

Good or bad requirements? (and why?)

- The system will enforce 6.5% sales tax on Washington purchases.
- The system shall display the elapsed time for the car to make one circuit around the track within 5 seconds, in hh:mm:ss format.
- The product will never crash. It will also be secure against hacks.
- The server backend will be written using PHP or Ruby on Rails.
- The system will support a large number of connections at once, and each user will not experience slowness or lag.
- The user can choose a document type from the drop-down list.

Good or bad requirements? (and why?)

- The system will enforce 6.5% sales tax on Washington purchases.
- The system shall display the elapsed time for the car to make one circuit around the track within 5 seconds, in hh:mm:ss format.
- The product will never crash. It will also be secure against hacks.
- The server backend will be written using PHP or Ruby on Rails.
- The system will support a large number of connections at once, and each user will not experience slowness or lag.
- The user can choose a document type from the drop-down list.

More examples ...

1. The system should be user friendly.
 2. The app must work on all devices.
 3. The system must handle a large number of user requests.
 4. The system should have an easy login.
-
- *The system shall allow a new user to complete the sign-up process in under 2 minutes without external assistance.*
 - *The app shall support the latest two major versions of iOS and Android on mobile devices.*
 - *The system shall support 10,000 concurrent users with an average response time below 1 second*

More examples ...

1. The system should have an easy login.

- *The system shall allow users to log in with a username and password within 3 attempts.*
- *The system shall support login via email/password and third-party providers (Google, Facebook).*
- *The system shall allow registered users to log in within 10 seconds on a standard internet connection.*
- *The system shall provide a 'Remember Me' option so users can log in with one click after the first authentication.*

WE'VE MADE PROGRESS. THE CUSTOMER'S
REQUIREMENTS HAVE GONE FROM BEWILDERING
TO VAGUE AND AMBITIOUS.

