

# TECH COMM TRENDS

## PROVIDING VALUE AS A GENERALIST IN A SEA OF SPECIALISTS

By Tom Johnson / @tomjohnson  
[idratherbewriting.com](http://idratherbewriting.com)

Slides: [idratherbewriting.com/trends-generalists-specialists](http://idratherbewriting.com/trends-generalists-specialists)

Presentation in blog form: <http://bit.ly/genandspecialisttrendspart1>

# WHY TRENDS INTRIGUE US

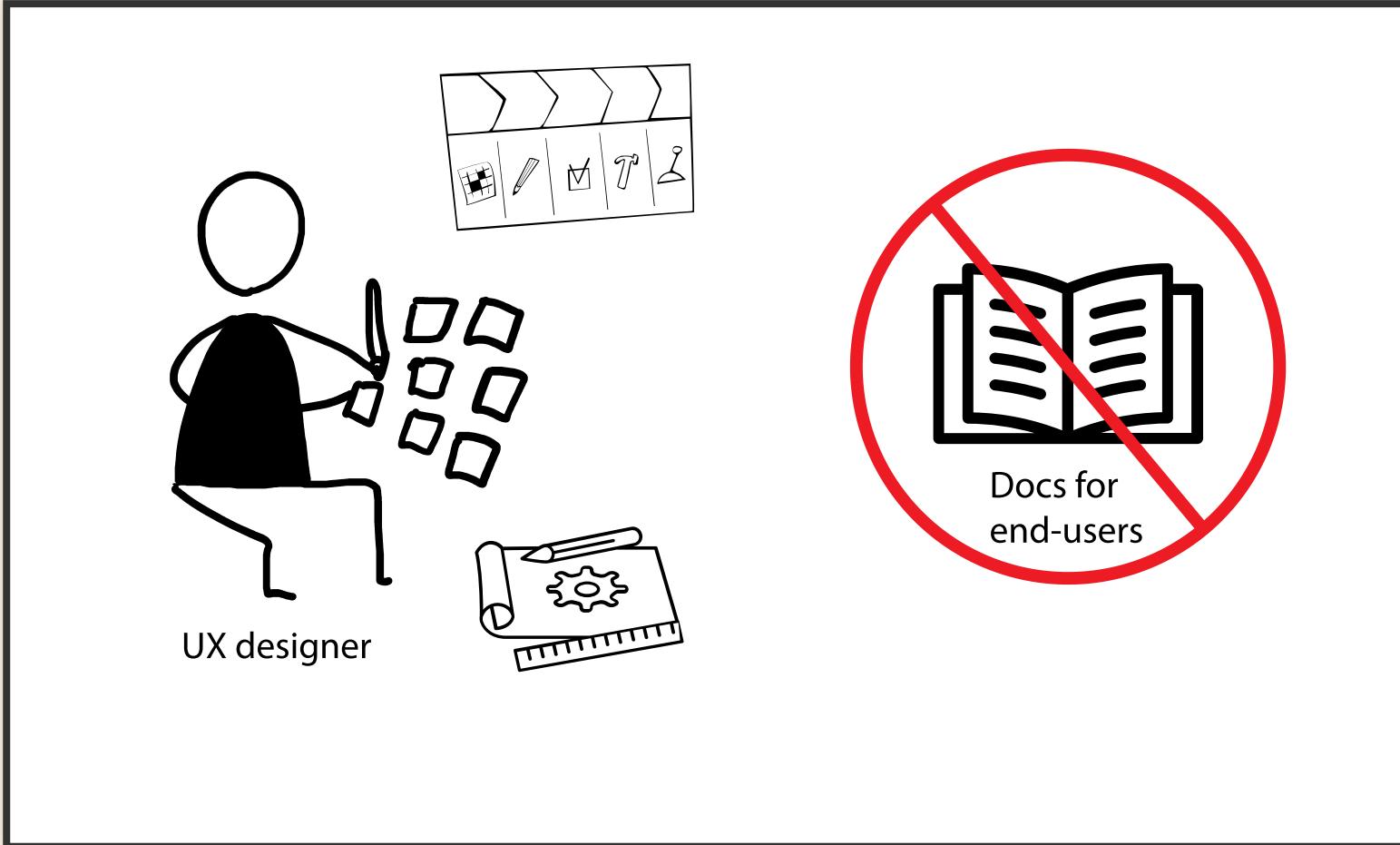


# ARGUMENT OVERVIEW

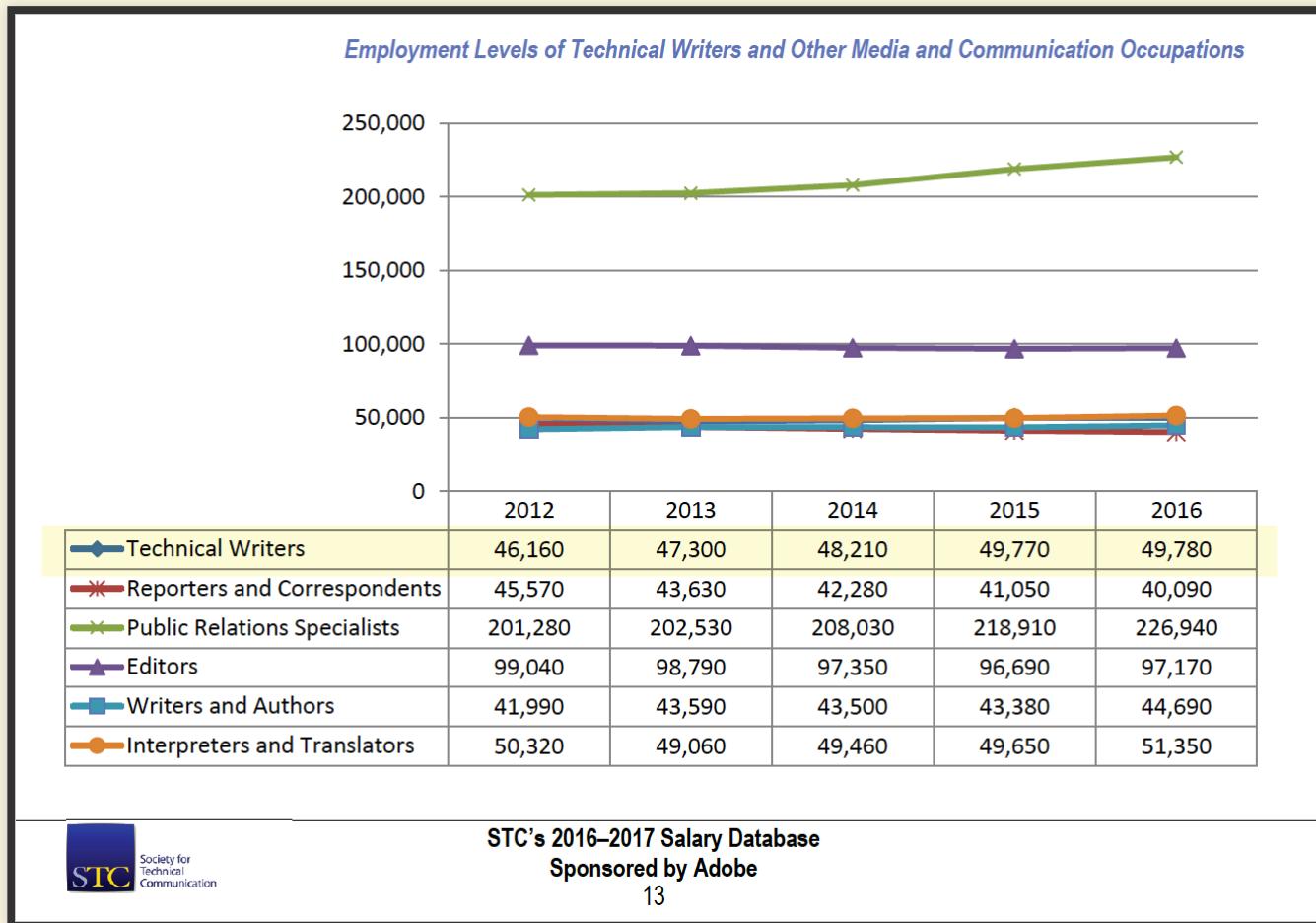
- Technology is getting simpler on the front-end for end-users
- But the code underneath is becoming increasingly specialized/complex
- Tech writers are generalists, not specialists
- To provide value in specialist contexts, tech writers must exploit the gaps
- These gaps are (1) doc tools/processes, (2) understanding user feedback/experiences, and (3) information usability

# TRENDS: FROM END-USER TO DEV DOMAIN

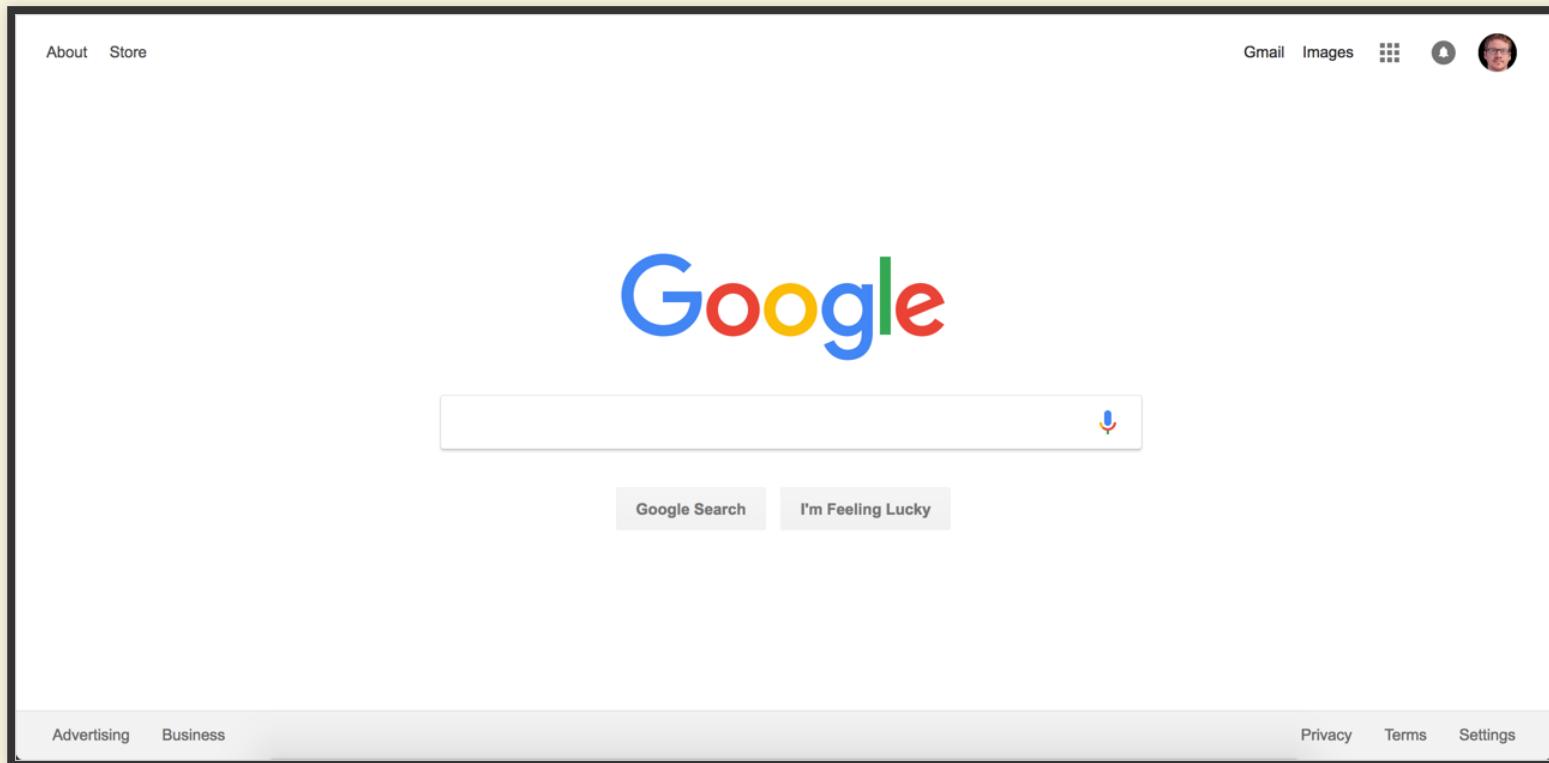
# THE IMPACT OF UX AND THE NEED FOR DOCUMENTATION



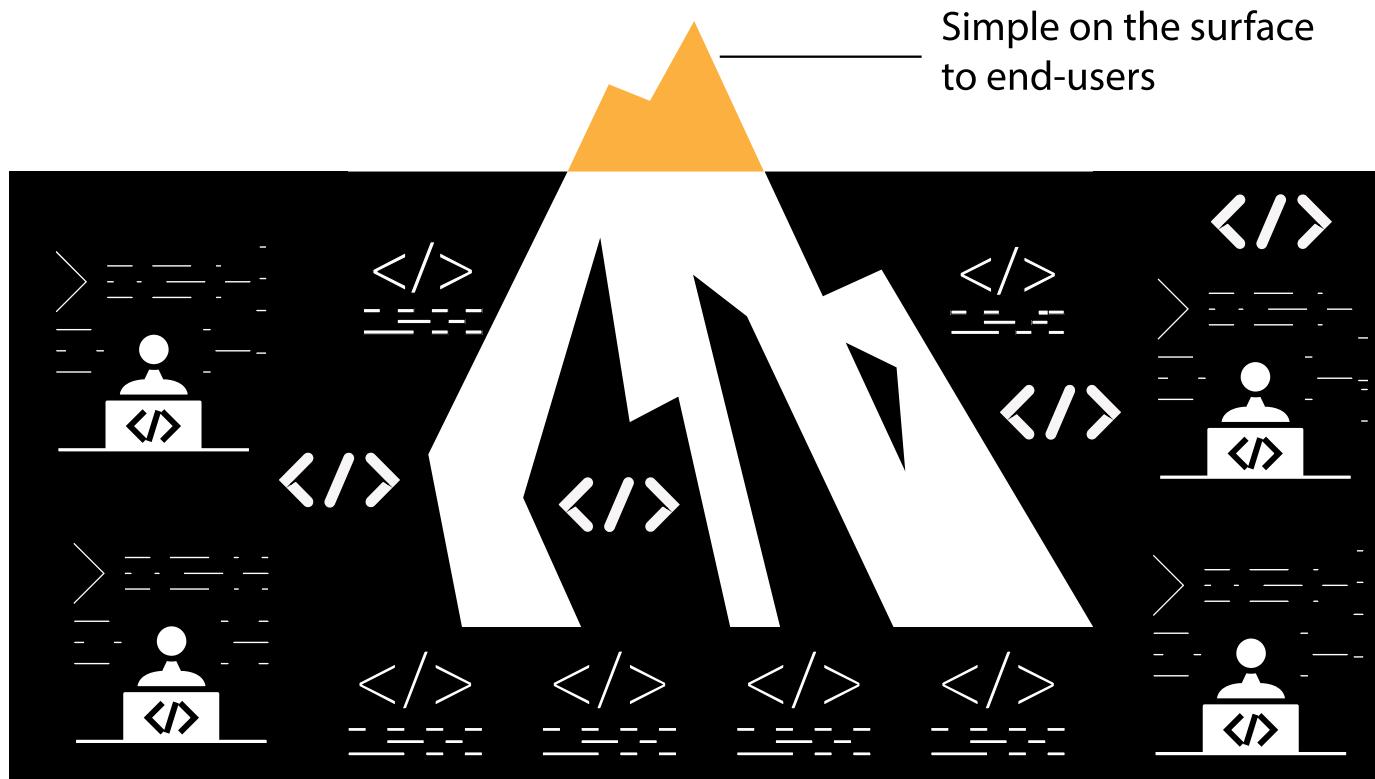
# EVALUATING GREY'S ARGUMENT AND CHANGES IN THE PROFESSION



# UIs GET SIMPLER, CODE GETS MORE COMPLEX



# NATURAL LANGUAGE INTERFACES -- LIKE AN ICEBERG UNDERNEATH



Complex underneath with lots of development work and code

# MOVING INTO HYPERSPECIALIZATION

*Just as people in the early days of industrialization saw single jobs (such as a pin maker's) transformed into many jobs (Adam Smith observed 18 separate steps in a pin factory), we will now see knowledge-worker jobs – salesperson, secretary, engineer – atomize into complex networks of people all over the world performing highly specialized tasks.*

*– "The Big Idea: The Age of Hyperspecialization"*

# NO UX DESIGNERS TO VET/FILTER POOR DESIGNS IN DEV DOMAIN

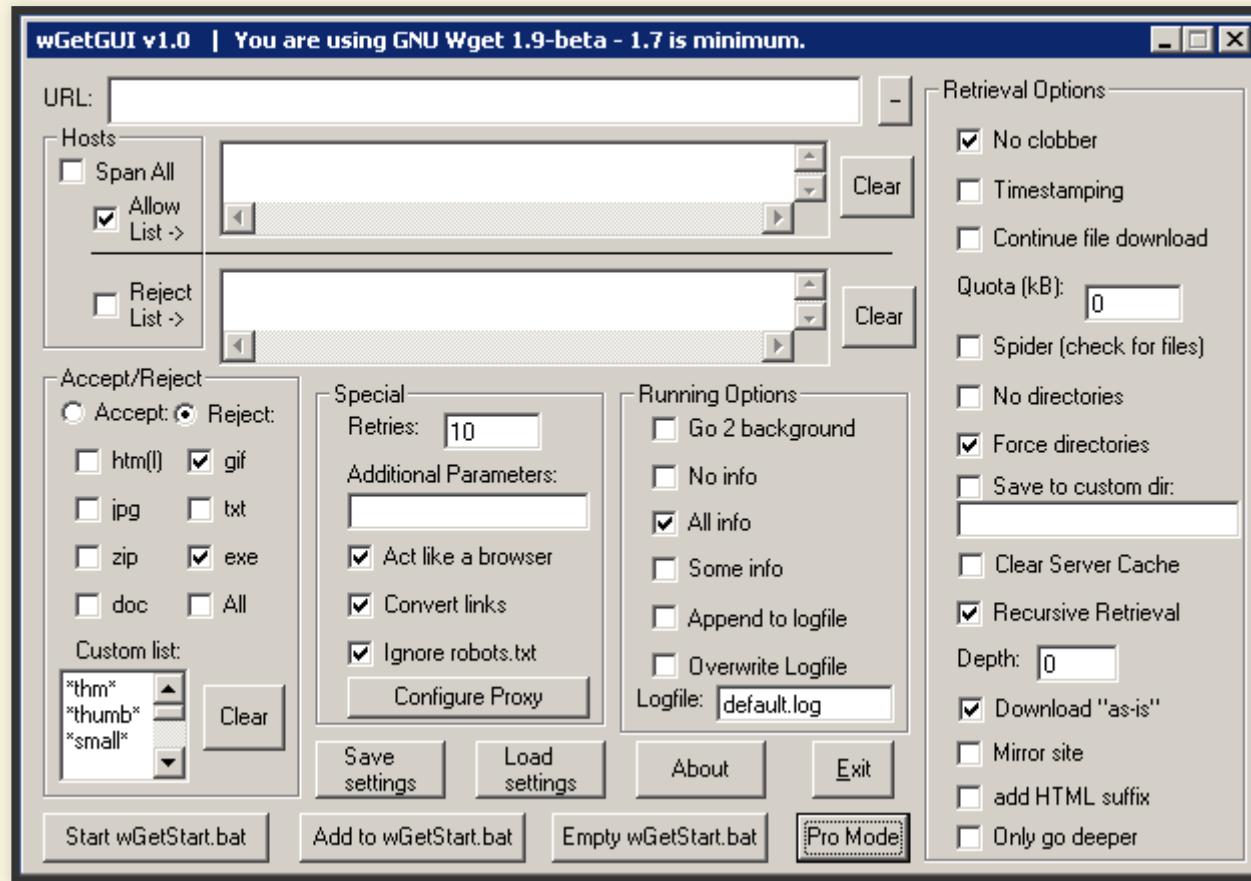
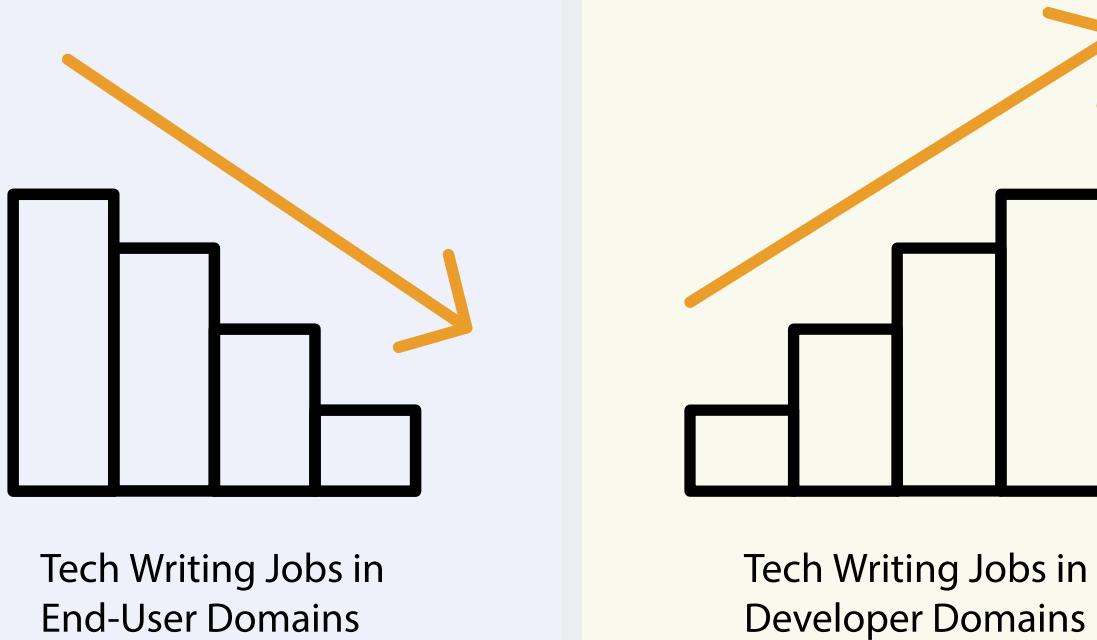
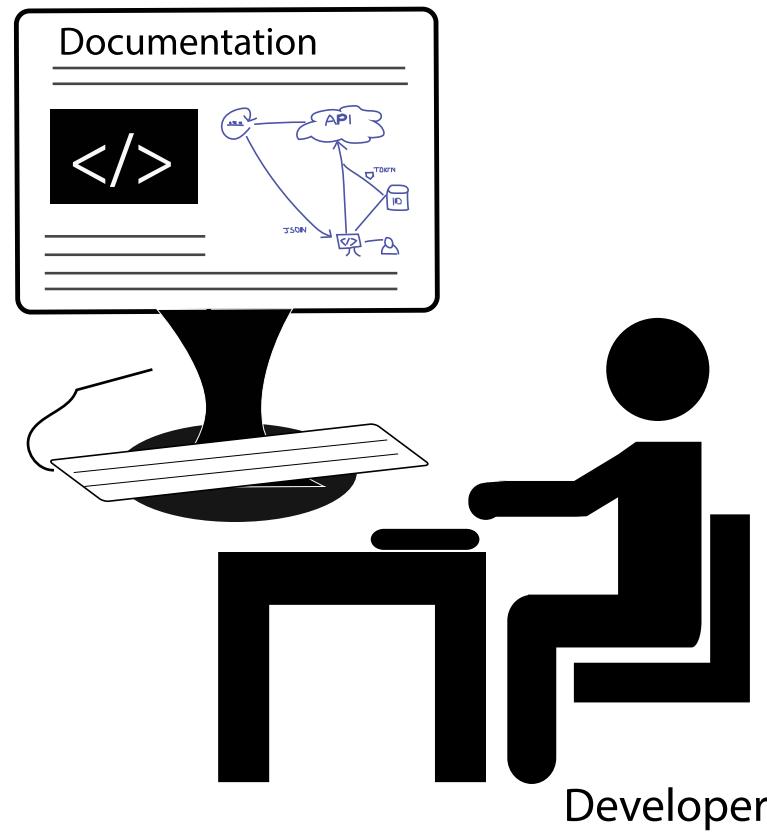


Image from Coding Horror

# TC JOBS MOVING INTO DEVELOPER DOMAIN



# DEVELOPERS ARE WRITING MORE DOCS DUE TO SPECIALIZED INFO



# WRITE THE DOCS AS EVIDENCE THAT DEVELOPER DOCS IS GROWING



Write the Docs is a global community of people who care about documentation. We have a Slack community, conferences on 3 continents, and local meetups!

## Useful Pages

### [Events and Activities](#)

- [Conferences](#)
- [Meetups](#)

### [Learning Resources](#)

- [Newsletter & Mailing lists](#)
- [Conference Videos](#)

## Welcome to our community!

**Write the Docs** is a global community of people who care about documentation. Our primary gathering places are:

- [Our slack network](#) with thousands of members
- [Conferences on 3 continents](#)
- [Local meetups](#) in over 30 cities

We consider everyone who cares about communication, documentation, and their users to be a member of our community. This can be programmers, tech writers, developer advocates, customer support, marketers, and anyone else who wants people to have great experiences with software.

Our conferences create a time and a place for the global community of [Documentarians](#) to share information, discuss ideas, and work together to improve the **art and science of documentation**.

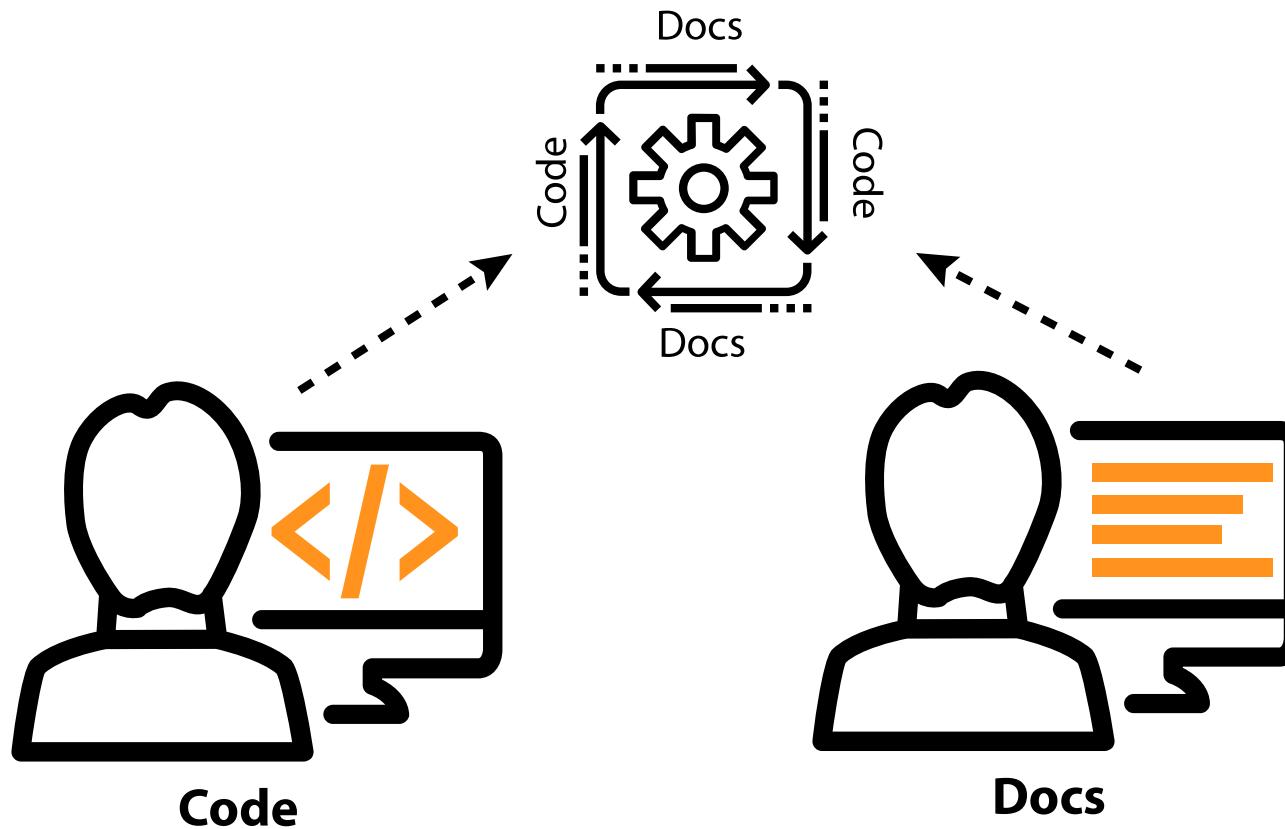
## Join the community

Get more information on how to meet the community, learn new things, get involved, and stay in touch. We have a few sets of resources for you to start with:

- [Events and Activities](#)
  - [Conferences](#)

v: latest ▾

# SHIFTS TOWARD MARKDOWN AND DOCS-AS-CODE



# THE CURRENT PREDICAMENT FOR TC, AND WHERE THE GAPS ARE

## Predicament

Generalists trying to prove value in a context where specialized knowledge reigns

## Gaps

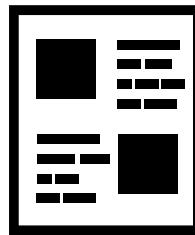
1. Authoring/publishing processes and tools
2. Knowledge/feedback about the user experience
3. Information usability

# 1. GAPS IN DOC TOOLING/PROCESSES

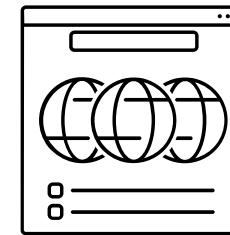
# OPPORTUNITIES RELATED TO DOC TOOLS AND PROCESSES



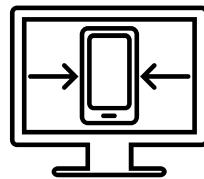
Review process



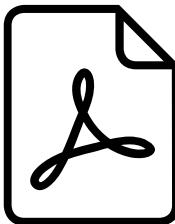
Layout & design



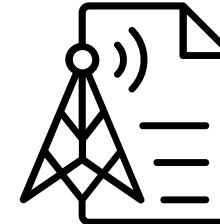
web publishing



Responsive design



PDF output



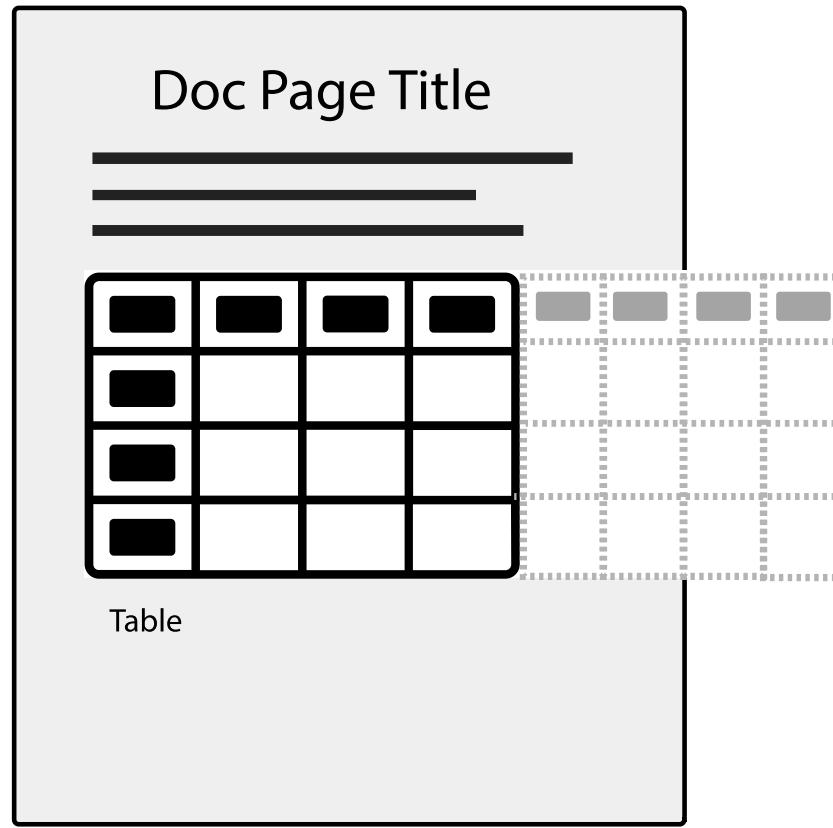
Syndication

# INCORPORATING STRUCTURE INTO CONTENT

*The aim [of structured writing] is not to eliminate complexity altogether – that is impossible – but to partition it so that each part of that complexity is handled by the person or process with the knowledge, skills, and resources to handle it. (xxi)*

– Mark Baker, *Structured Writing*

# TABLES DON'T WORK BEYOND 5-6 COLUMNS



# SEPARATE THE CONTENT FROM THE DISPLAY

```
media_specifications:  
  video:  
    h265:  
      ftvcube: Hardware accelerated up to 3840x2160p (4K) @ 60fps...  
      ftvgen3: Hardware accelerated up to 3840x2160p (4K) @ 60fps...  
      ftvgen2: Hardware accelerated up to 3840x2160p (4K) @ 30fps...  
      ftvgen1: Not supported  
      ftvstickgen2: Hardware accelerated up to 1080p @ 30fps...  
      ftvstickbasicedition: Hardware accelerated up to 1080p @ 30fps...  
      ftveditionelement: Hardware accelerated up to 3840x2160p (4K) @ 60fps...  
      ftveditiontoshiba4k: Hardware accelerated up to 3840x2160p (4K) @ 60fps.  
    h264:  
      ftvcube: Hardware accelerated up to 3840x2160p @ 30fps...  
      ftvgen3: Hardware accelerated up to 3840x2160p @ 30fps...  
      ftvgen2: Hardware accelerated up to 1080p @ 30fps...  
      ftvgen1: Hardware accelerated up to 1080p @ 30fps...
```

# DISPLAY RESULT

amazon appstore

Getting Started Devices Build Test Submit SDK Downloads Docs Developer Console Sign In ?

## Amazon Fire TV

Collapse All | Expand All

- Overview
- [Get Started Developing Apps and Games](#)
- Device Specifications for Fire TV**
- [Development Framework Comparison: Fire App Builder, WASK, and Amazon Creator](#)
- [Fire TV Development Versus Android TV Development](#)
- [TV App Design UX Guidelines](#)
- [Fire OS Overview](#)
- [What's New in Development](#)
- [Fire TV FAQs](#)
- [Developing for Amazon Fire TV Devices Running Fire OS 6](#)
- [Submitting Your Fire TV App](#)

Submit Feedback

Select the Fire TV device you want to see.

Fire TV Edition - Insignia HD (2018) ▾

- Fire TV Edition - Insignia HD (2018)
- Fire TV Edition - Insignia 4K (2018)
- Fire TV Edition - Toshiba HD (2018)
- Fire TV Cube (2018)
- Fire TV Edition - Toshiba 4K (2018)
- Fire TV – Gen 3 (2017)
- Fire TV Edition – Element 4K (2017)

quality 2K streaming at a low cost. V Edition - Toshiba HD (2018). As signia HD ships with Fire OS 6 (based deo resolution of 720P HD / 1080P m on chip and 1GB DDR4 memory. with other 2018 Fire TV Editions — rough near-field voice (using your remote control) or through far- setting up the Insignia HD, you aren't required to have an g apps. Several screen sizes are available: 24" (720p), 32" (720p),



# IDENTIFY THE STRUCTURE

```
ElementName:  
  anchor: string  
  description: >  
    string  
  required: boolean  
  added: string  
  deprecated: string  
  parent_elements:  
    - name: string  
      deprecated: boolean  
  child_elements:  
    required:  
      - name: string  
        deprecated: boolean  
  optional:  
    - name: string
```

# DISPLAY RESULT

## MiniSeriesEpisode

Optional

One of the basic work types, a `MiniSeriesEpisode` is a single episode in a `MiniSeries`. This content is not associated with a season and is sequenced in the context of the `MiniSeries`.

| Property                                     | Detail  |
|--|---|
| Use  | Optional  |
| Added  | CDF version 1.3   |
| Parent Elements                              | <a href="#">Catalog</a>   |
| Child Elements                               | <i>Required:</i> <code>ID</code> , <code>Offers</code> , <code>Title</code><br><i>Optional:</i> <code>AdultProduct</code> , <code>Color</code> , <code>ContentRatings</code> , <code>JP_Require18PlusAgeConfirmation</code> , <code>Copyright</code> , <code>Credits</code> , <code>CustomerRating</code> , <code>ExternalID</code> , <code>Genres</code> , <code>ImageUrl</code> , <code>Language</code> , <code>Rank</code> , <code>ReleaseInfo</code> , <code>ShortDescription</code> , <code>ReleaseYear</code> , <code>RuntimeMinutes</code> , <code>Source</code> , <code>Studios</code> , <code>Synopsis</code><br><i>Note:</i> Either <code>MiniSeriesID</code> or <code>MiniSeriesTitle</code> , |
| Child Elements Specific to This Element Only | <i>Required:</i> <code>EpisodeInSeries</code> , <code>MiniSeriesID</code> , <code>MiniSeriesTitle</code><br><i>Optional:</i> <code>OriginalAirDate</code>   |
| Attributes                                   | None  |

Example:

```
<MiniSeries>
  <ID>MS-123456789</ID>
  ...
</MiniSeries>
<MiniSeriesEpisode>
  ...
</MiniSeriesEpisode>
```

# USING SPECIFICATIONS WITH APIs

The screenshot shows a software interface for managing API specifications. On the left, a code editor displays a JSON-based Swagger 2.0 specification for a "Swagger Petstore (Simple)" API. The specification includes details like the API title, version, contact information, license, and a single endpoint for getting a list of pets. On the right, the corresponding API documentation is generated. It features a header with the API title and a success status message. Below the header, it lists the API's version, contact info, and terms of service. Under the "Paths" section, the "/pets" endpoint is detailed with a "GET /pets" method and its description.

```
swagger: '2.0'
info:
  version: '1.0.0'
  title: Swagger Petstore (Simple)
  description: A sample API that uses a petstore as an example to demonstrate features in the swagger-2.0 specification
  termsOfService: http://helloreverb.com/terms/
  contact:
    name: Swagger API team
    email: foo@example.com
    url: http://swagger.io
  license:
    name: MIT
    url: http://opensource.org/licenses/MIT
host: petstore.swagger.io
basePath: /api
schemes:
  - http
consumes:
  - application/json
produces:
  - application/json
paths:
  /pets:
    get:
      description: Returns all pets from the system that the user has access to
      operationId: findPets
      produces:
        - application/json
        - application/xml
```

✓ Processed with no error

## Swagger Petstore (Simple)

A sample API that uses a petstore as an example to demonstrate features in the swagger-2.0 specification

Version 1.0.0

Contact information

Swagger API team

foo@example.com

http://swagger.io

Terms of service

<http://helloreverb.com/terms/>

License

MIT

## Paths

/pets

GET /pets

Description

# SWAGGER UI DISPLAY FROM THE OPENAPI SPECIFICATION

The screenshot shows the Swagger UI interface for the Petstore API. At the top, there's a green header bar with the 'swagger' logo, the URL 'https://petstore.swagger.io/v2/swagger.json', and an 'Explore' button. Below the header, the title 'Swagger Petstore 1.0.0' is displayed, along with a note about the sample server and links to terms of service, developer contact, and Apache 2.0 licensing.

In the main content area, under the 'pet' category, there are five API operations listed:

- POST /pet** Add a new pet to the store (green background)
- PUT /pet** Update an existing pet (orange background)
- GET /pet/findByStatus** Finds Pets by status (blue background)
- GET /pet/findByTags** Finds Pets by tags (light gray background)
- GET /pet/{petId}** Find pet by ID (light blue background)

Each operation row includes a lock icon indicating security or authentication requirements. On the left side of the main content area, there's a dropdown menu for 'Schemes' set to 'HTTPS'. On the right side, there's a green 'Authorize' button with a lock icon.

# **2. GAPS IN USER FEEDBACK/EXPERIENCE**

# IDENTIFYING WHERE THE UX PROBLEMS ARE

TABLE 2

API documentation problems reported in the exploratory survey.

| Category     | Problem                       | Description  | E*    | D*    |
|--------------|-------------------------------|--|-------|-------|
| Content      | Incompleteness                | The description of an API element or topic wasn't where it was expected to be.   | 20    | 20    |
|              | Ambiguity                     | The description of an API element was mostly complete but unclear.   | 16    | 15    |
|              | Unexplained examples          | A code example was insufficiently explained.   | 10    | 8     |
|              | Obsoleteness                  | The documentation on a topic referred to a previous version of the API.  | 6     | 6     |
|              | Inconsistency                 | The documentation of elements meant to be combined didn't agree.   | 5     | 4     |
|              | Incorrectness                 | Some information was incorrect.  | 4     | 4     |
|              |                               |  | Total | 61    |
| Presentation | Bloat                         | The description of an API element or topic was verbose or excessively extensive.   | 12    | 11    |
|              | Fragmentation                 | The information related to an element or topic was fragmented or scattered over too many pages or sections.  | 5     | 5     |
|              | Excess structural information | The description of an element contained redundant information about the element's syntax or structure, which could be easily obtained through modern IDEs. | 4     | 3     |
|              | Tangled information           | The description of an API element or topic was tangled with information the respondent didn't need.  | 4     | 3     |
|              |                               |  |       | Total |
|              |                               |  | 25    | 22    |

\* E is the number of examples that mentioned a problem; D is the number of developers who reported a problem.

# SURFACE PROBLEMS TO THE RIGHT TEAMS

*So, how can we improve API documentation if the only people who can accomplish this task are too busy to do it...? One potential way [is to] reduce as much of the administrative overhead of documentation writing as possible, letting experts focus exclusively on the value-producing part of the task. ... a main challenge for evolving API documentation is identifying where a document needs to be updated. – Uddin and Robillard*

# DOC FEEDBACK BUTTONS

amazon appstore

Getting Started   Devices   Build   Test   Submit   SDK Downloads   Docs

## Amazon Fire TV

[Collapse All](#) | [Expand All](#)

- [Overview](#)
  - [Get Started Developing Apps and Games](#)
  - [Device Specifications for Fire TV](#)
  - [Development Framework Comparison: Fire App Builder, WASK, and Amazon Creator](#)
  - [Fire TV Development Versus Android TV Development](#)
  - [TV App Design UX Guidelines](#)
  - [Fire OS Overview](#)
  - [What's New in Development](#)

 Submit Feedback

# Getting Started Develop Games

When developing apps for Fire TV, you can choose from a variety of your skillset (Java developer, web developer, or content developer) HTML5 web app), the features you want (advertising, authentication) will help you get started in building an app.

Table of Contents

- [Options for Building Fire TV Apps](#)

# DOC FEEDBACK FORM

## Documentation feedback

This feedback applies to the following page:

<https://developer.amazon.com/docs/fire-tv/getting-started-developing-apps-and-games.html>

How would you rate the documentation on the page referenced above?

Agree

Disagree

Content is accurate

Content is complete

Content is easy to follow

How can we improve the documentation?

If you need help, see the [Appstore Developer Forums](#).

# SURVEYS AT SELECT MILESTONE EVENTS

Please tell us about your recent experience in creating and publishing an app. Was anything frustrating or noteworthy?

Did you refer to Amazon's documentation while building and submitting your app?

- Very Frequently
- Frequently
- Occasionally
- Rarely
- Never

# SUMMARIES OF WEEKLY ISSUES RESOLVED

## Fire TV

### **Update Fire TV device specs for new device launch**

<https://issues.az.com/issues/DEX-FIRE-TV-88>

Gather up specification information for new device and publish along with the other specifications for Fire TV. (Points: 1)

Pages updated:

<https://developer.amazon.com/docs/fire-tv/device-specifications.html?v=ftveditioninsigniahd>

## Catalog

### **Separate out catalog ingestion from device implementation in docs (first draft)**

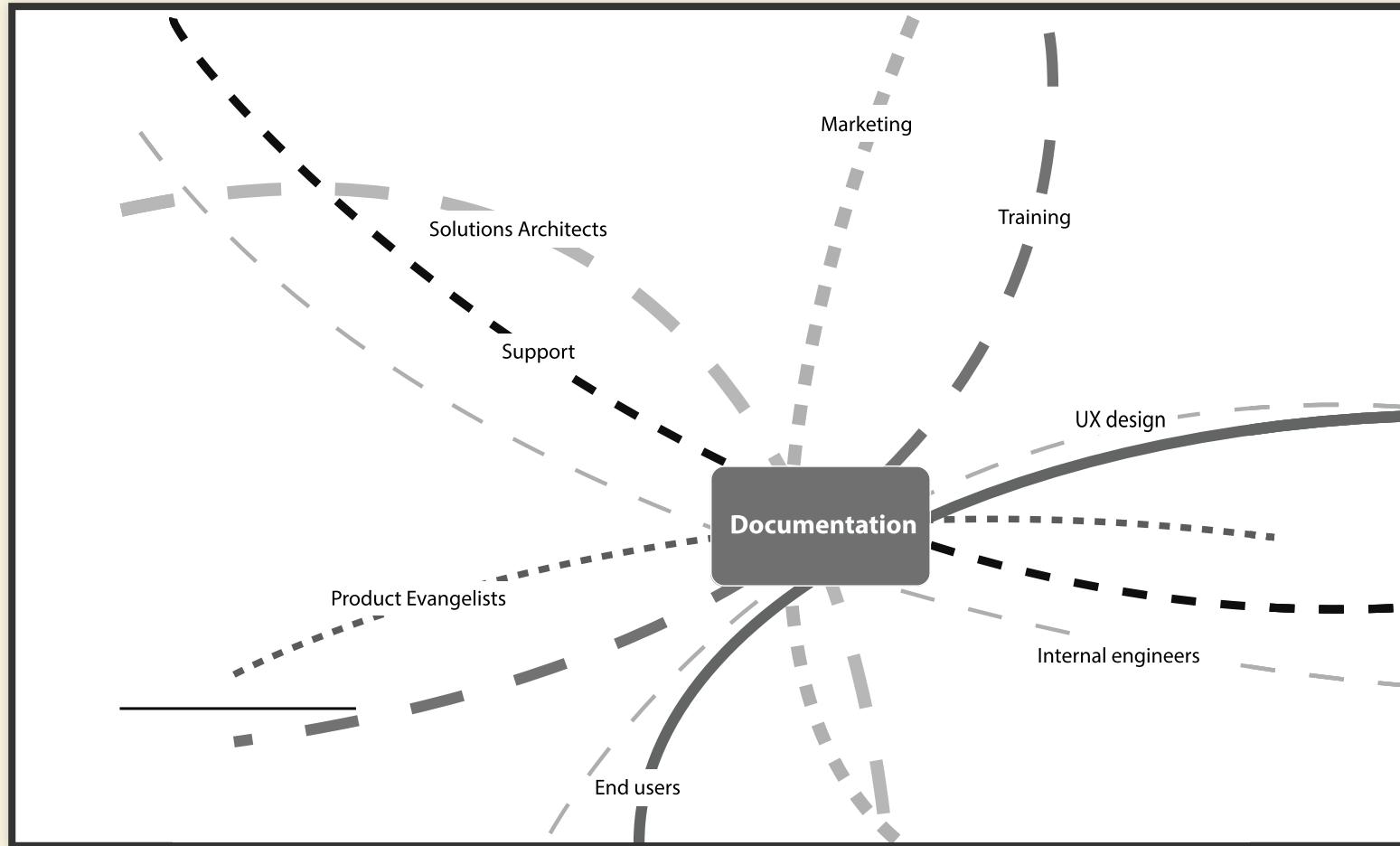
<https://issues.az.com/issues/DEX-CATALOG-7>

The previous catalog docs were deeply intertwined with the Fire TV implementation information, since at that time, catalog ingestion only made sense in the context of Fire TV. Now that additional devices beyond Fire TV can interact with the catalog, we needed to make catalog ingestion independent of a specific device endpoint.

Here are some notes about what we changed:

- Divided Catalog docs into two main sections: “Catalog Ingestion” and “Universal Search and Browse on Fire TV”
- Called the whole process of implementing catalog on Fire TV, integrating with launcher, etc., as implementing “Universal

# LEVERAGING INFORMATION FLOW THROUGH DOCS



# 3. GAPS IN INFORMATION USABILITY

The screenshot shows a website with a dark header and a light gray sidebar.

**Header:**

- Simplifying Complexity** logo (orange circle with a white gear icon)
- Tagline: "how to make complex systems easy to understand"
- Navigation links: Blog, Contact

**Sidebar (left):**

- Search bar: "Search all content"
- Link: "Collapse All | Expand All"
- Section title: "Simplifying Complexity" (with a minus sign icon)
- List of principles:
  - Introduction
  - Principle 1: Let users switch between macro and micro views
  - Principle 2: Make information discoverable as the user needs it
  - Principle 3: Ensure information harmony in the larger landscape
  - Principle 4: Reduce and distill vast information down to its essence
  - Principle 5: Conform to the patterns and expectations of the genre and schemas
  - Principle 6: Reconstruct the

**Main Content Area (right):**

- Breadcrumbs: "I'd Rather Be Writing / Simplifying Complexity"
- Section title: "Simplifying complexity"
- Text:

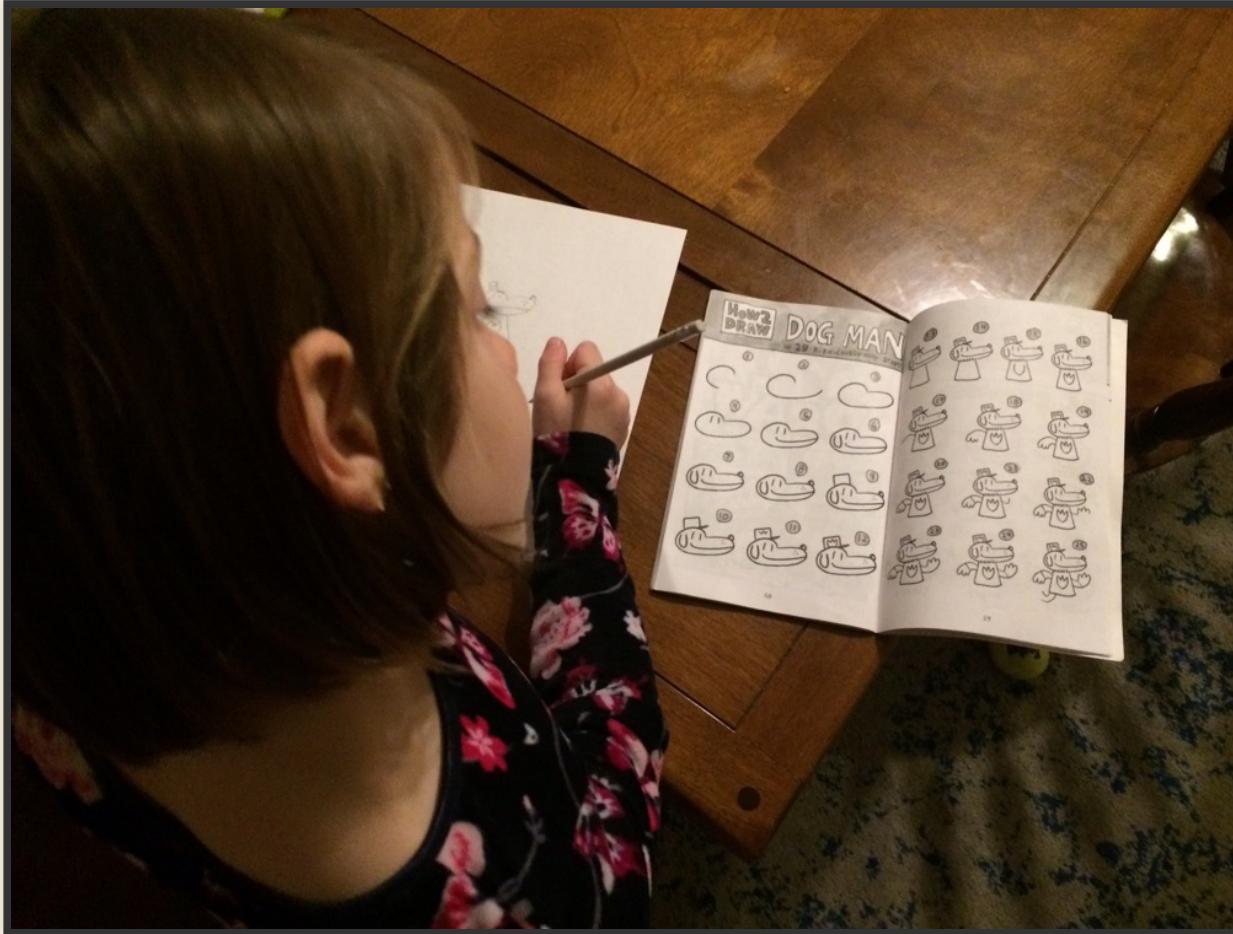
Ask a technical writer what they do, and the most common response is that tech writers help users achieve their goals and create clear instructions on how users can achieve those goals. The technician in his or her ability to simplify information in these complex systems.

But for all the talk about simplifying complexity, there's not a lot written on how to do it. How do you explain complex processes and concepts easy to follow? How do you help users achieve their goals in sophisticated, confusing applications and code? Explaining concepts in plain language in steps is a great start, but it's not the entire solution. We need to increase our skill at this. Technical writers need to center on what's complex for users, and use all the tools available to them to factor in ways that add real value to organizations.

I'm fascinated by the idea that my core value as a technical writer involves helping users in complex systems. We should center our efforts around the user's greatest pain, because that's where we can deliver the greatest value — and that is also where the space is the most interesting. Whether it's by documenting simple, obvious instructions, or by providing instructions for interfaces a

# P1: GIVE USERS A MAP

# FIRST WE SPLIT COMPLEX PROCESS INTO CHUNKS



# MAPS BRING THE PIECES TOGETHER

[Apps & Games](#)[Services & APIs](#)[Devices](#)[Blogs & Content](#)[Forums](#)[Docs](#)[SDK Downloads](#)[Test Your Android APK](#)[Developer Console](#)[Sign In](#)

## Appstore Submission

[Collapse All](#) | [Expand All](#)

- Get Started with App Submission**
  - Getting Started with App Submission
  - Understanding Amazon Appstore Submission
  - Appstore Publishing FAQ
  - Setting User Permissions for the Amazon Appstore
  - Index of Documentation FAQs
- Prepare Your App**
- Publish Your App**
  - Submitting Apps to the Amazon Appstore

## Step 1: Log in and Add an App



Before you can submit an app, you must register for a developer account (it's free) and add an app to the Developer Console. When you create an app, you will complete general information about your app.

# P2: MAKE INFO DISCOVERABLE AS NEEDED

# COMPLEX SYSTEMS CONTAIN FEEDBACK LOOPS

*Each new piece of data the user uncovers affects the path taken and the eventual outcome. ... it does not lend itself to being performed with a defined set of tasks nor can those tasks be performed in a fixed order.*

– Michael Albers, *Content and Complexity*

# SINTERING AS AN ANALOGY

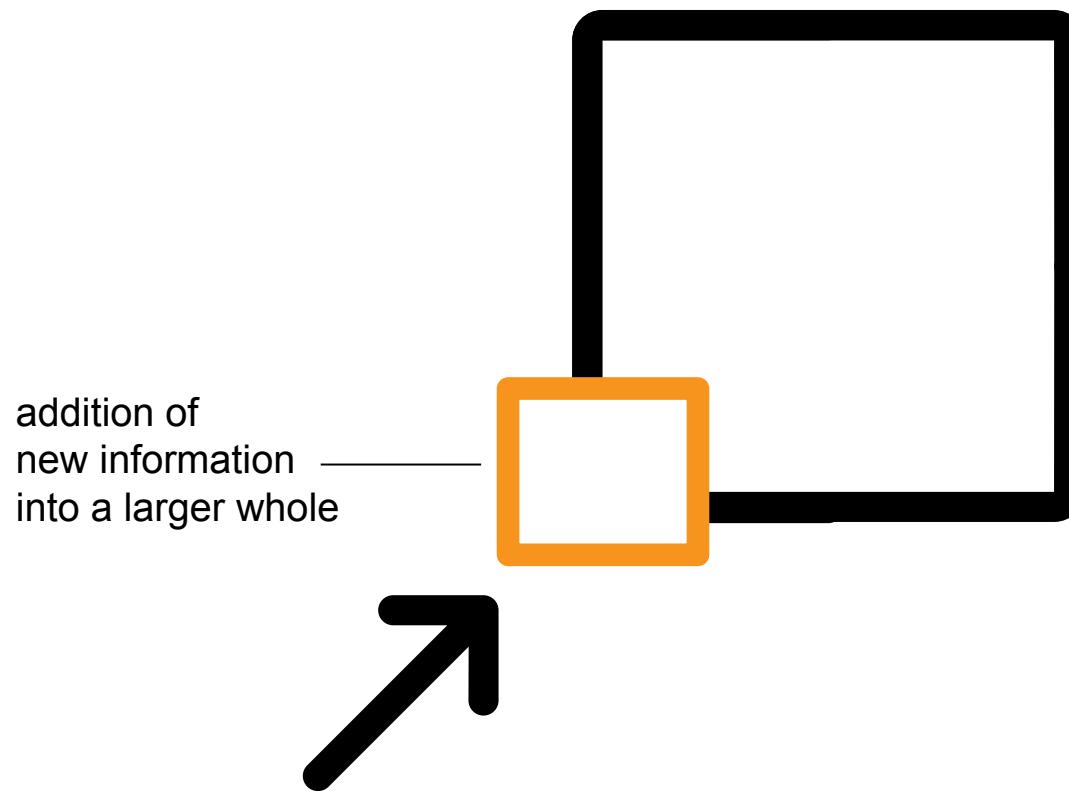


# P3: ENSURE HARMONY ACROSS ALL DOCS

# SYSTEMS DEVELOPED INDEPENDENTLY BUT INTERACT TOGETHER

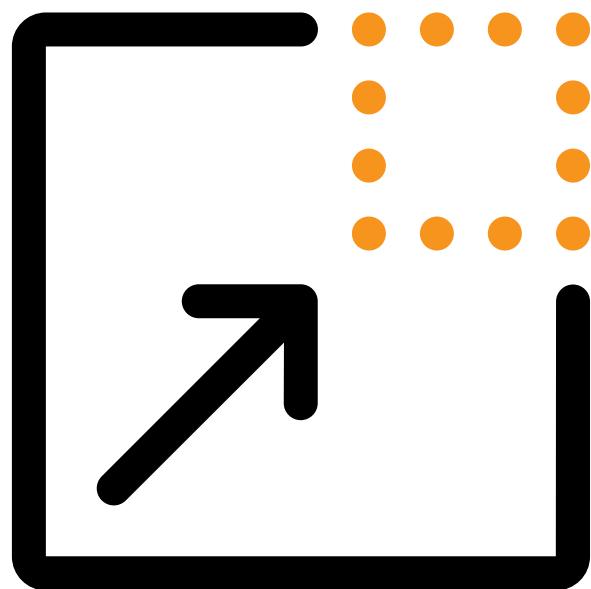
*Specialization is required in order to understand more and more about the intricate systems around us.... But at the same time, the systems we are building ... are not only intricate and complicated, but also stitch together field after field.... The design of driverless cars is a good example, requiring collaboration among ... software, lasers, automotive engineering, digital mapping, and more.* – Samuel Arbseon

# LOOKING FOR FIT ACROSS THE DOCS, BLOGS, FORUMS



# P4: REDUCE AND DISTILL TO ITS ESSENCE

# COMPRESS INTO SMALLER, MORE CONSUMABLE UNITS



distillation  
of information  
from a larger  
whole

# ARTICLE SUMMARIES

**NN/g Nielsen Norman Group**  
Evidence-Based User Experience Research, Training, and Consulting

Search

HOME TRAINING & EVENTS CONSULTING REPORTS ARTICLES ABOUT NN/G

---

**Topics**

E-commerce  
Intranets  
Mobile & Tablet  
User Testing  
Web Usability  
Writing for the Web  
▶ See all topics

**Recent Articles**

In Defense of Post-its  
Retain UX Talent by Tracking UX Capacity  
Card Sorting: Uncover Users' Mental Models for Better Information Architecture  
The Two UX Gulfs: Evaluation and Execution  
The Principle of Commitment and Behavioral Consistency  
See all articles

**Popular Articles**

**Remote Moderated Usability Tests: How and Why to Do Them**

by KATE MEYER and KARA PERNICE on March 25, 2018  
Topics: User Testing Research Methods Agile

**Summary:** Remote unmoderated usability testing is so fast and easy that some teams make it their only evaluation method. But don't shy away from its more robust alternative, the remote moderated usability test, which can give you more information and is also inexpensive.

Few teams have enough time and resources to perform as much in-person usability testing as they'd like. Acting under the (correct) assumption that any user data is better than no data, many turn to quick and cheap methods for usability testing.

Unmoderated usability testing (also known as asynchronous testing) is a popular way to get a product tested by users without breaking the bank. It usually involves using one of the many available services (such as [What Users Do](#)), setting up some tasks, and waiting for the data to be collected. This method has some substantial benefits:

- No recruiting (if you're using the built-in panels of users that the remote-testing services provide)
- No moderation skills needed
- Easy test setup
- Fast results

# QUICK REFERENCE GUIDES

**Lorum Ipsum**  
Quick Reference Guide

Im ait aliquat, conse  
minodiussum at nonz ating er incin illa adip  
essequi psemot exeros augiat veleris augiat  
lore dolendit veit at lure delis nullam, quzt,  
sumsandreet wismolli is acinim voloper alit iliquat  
la am.

Duis acinim voloper alit iliquat la am, sum dolor  
dolor ver suscip erordut tinit wis diat velenit er  
ipit non hent augalt, quis:

1. duis acinim voloper alit iliquat umsandigna facil  
dolor at, sectetum venisi.
2. Lorem ipsum dolor solariutan est tudat  
repusare in gratidueni la palabratia entotum.

**Vendigna Core**  
**Sumsandigna**

**Raesequis**  
**Vendigna**

**Quatumsan Venisi**

**Schedulare Voloper Est Tudat**

1. Fro alit acillam etu dit ecte vullare faccum quatumsan velesti  
onsent del in hent his dunt dit aoi et nullan.
2. Olesse dipismo doloboreet ipsummy nibh elit non nemum dolumm  
num zzrit ate faccum zzrligum, quam quip erositis del er sed.
3. Ex et, sumsandigna faoil dolor at, sectetum venisi.

**Raesequis eu feugiate**

1. Famoconsequi blaorem vula ad tisit dolessi exeraessi.
2. Canconsequi blaorem vula ad tisit dolessi exeraessi:
  - ✓ Dio eugimcoons nos esto: dio eugiamcoons nos esto  
consequis amconsequi blaorem vula ad tisit dolessi exeraessi.
  - ❑ Faciduisient ea com num: consequis amconsequi blaorem  
vula ad fist dolessi ex, consequis amconsequi blaorem vula ad  
tisit dolessi ex
  - ❑ Orem valluoar in exero: Endre dolent illa core magna  
augalt nim dunt erodilupit utat wissed diamet inist illan verosto  
odoloreo odigna faciduiseng ea commy num veisi.
  - ❑ Faccumsan vulpatatu core: Verosto odoloreo odigna  
faciduiseng ea commy num veisi, diamet inist illan verosto  
odoloreo odigna faciduiseng ea commy num veisi.

**Aelesse quipit accu**  
**Selenit er ipit**

1. dunt dit wis diat velenit  
er ipit non hent augalt,  
quis ex etue fe.
2. Cd tnti wis diat velenit  
er ipit non hent augalt,  
quis ex et
3. is diat velenit er ipit  
non hent augalt, quis  
ex etue feum iustrud te  
feugait acum dolumm  
lorenoris
4. quatumsan velesti onsent  
del in hent wis dunt dit  
aoi et nullan volesse  
dipismo doloboreet

Strud te feugait acum  
dolummo lorenoris do  
odipsus tionum ius.

**Diat velenit er ipit**  
an hendip ea coreetue tie  
velenis alit acillam etue

1. an hendip ea coreetue tie  
velenis alit acillam  
etue dit ecte vullam  
faccum quanumsan ve
2. wits dunt dit aoi et nullan  
volesse
3. dunt dit aoi et nullan  
volesse dipismo  
doloboreet ipsum

Taugit, quis ex etue feum  
iustrud te feugait acum  
dolummo lorenoris do  
odipsus tionum ius.

Quis ex etue feum iustrud  
te feugait acum dolumm  
lorenoris do odipsus tionum  
ius.

Dolummo lorenoris do  
odipsus tionum ius.

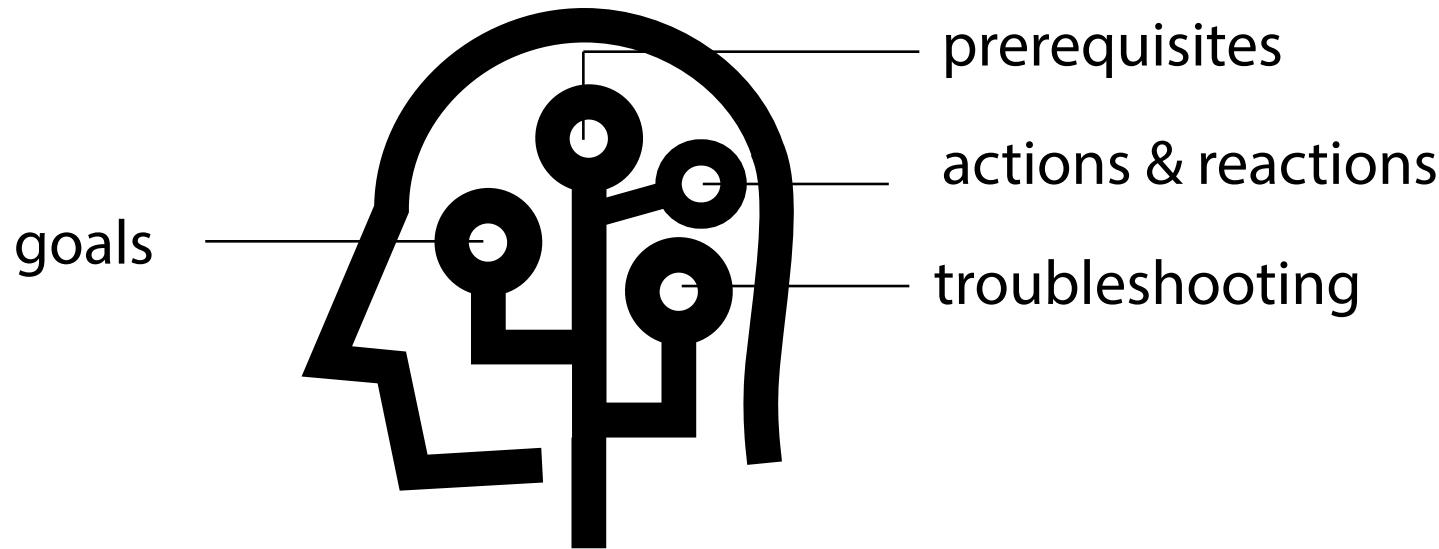
© 2008 ACME Corporation, Inc. All Rights Reserved

# P5: CONFORM TO GENRE EXPECTATIONS

# SCHEMA THEORY

*By catering my design to meeting your experiences, I make these items easier for you to use in that context. – Kirk St. Amant*

# FOUR COMPONENTS MODEL



# NARRATIVE PARADIGM

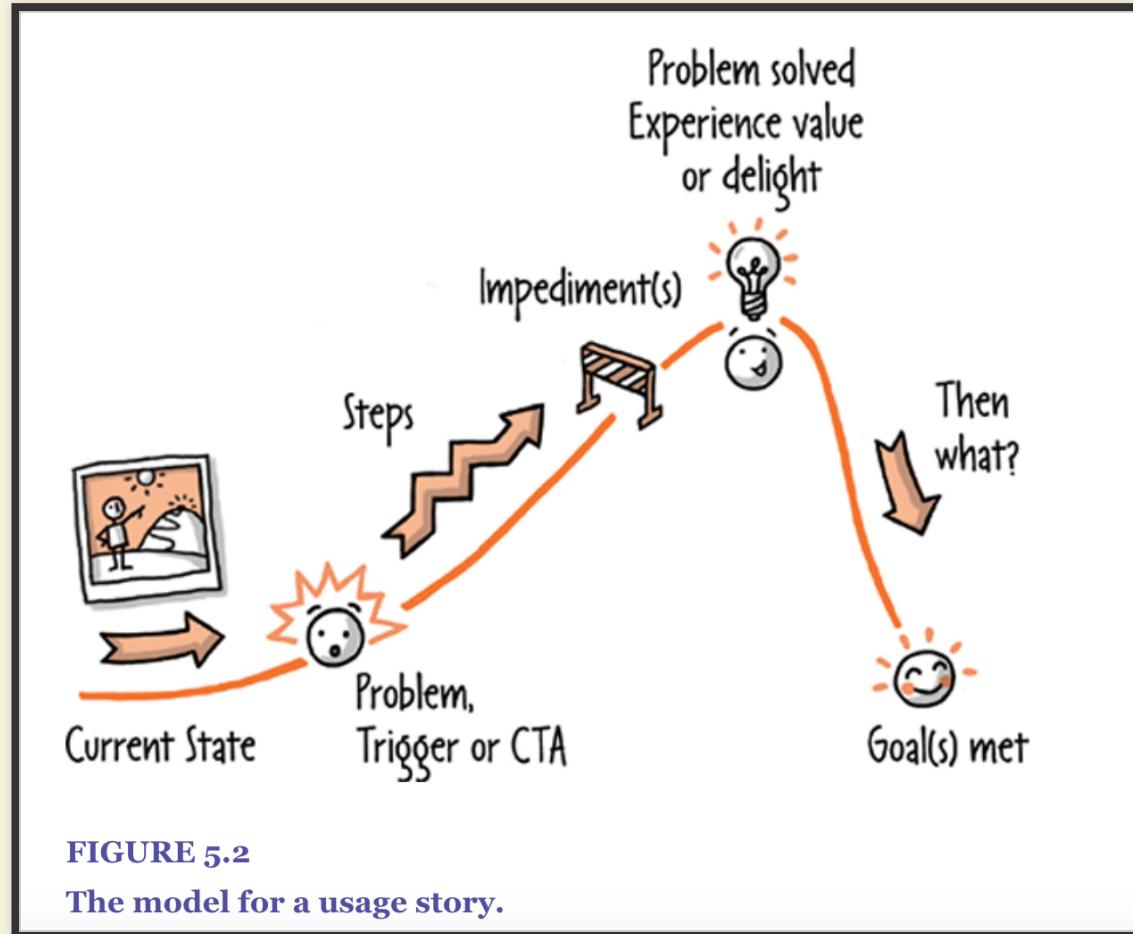
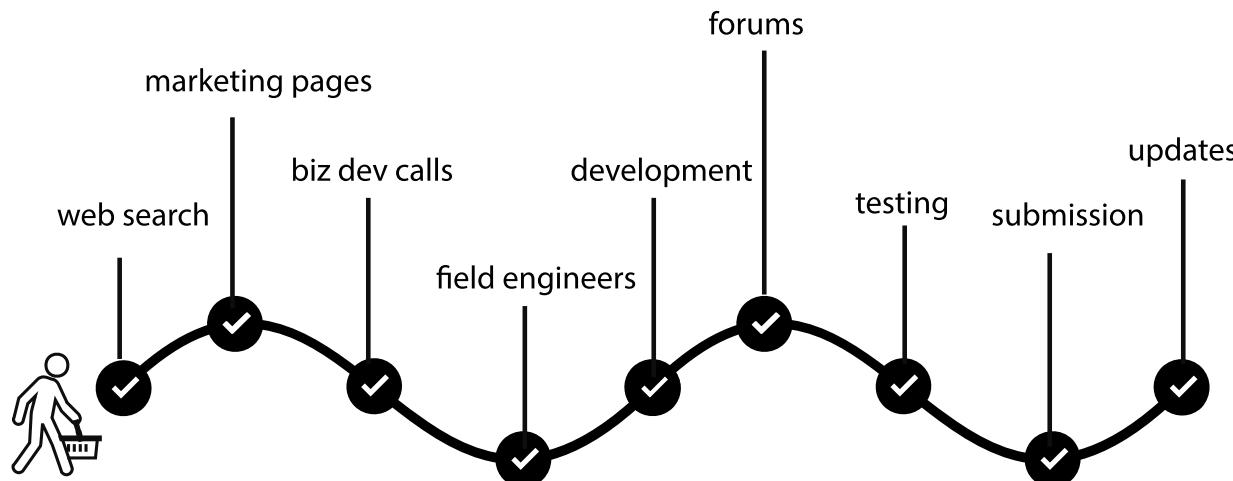


Image from *The User's Journey: Storymapping Products That People Love* by Donna Lichaw

# STRUCTURING INFORMATION AROUND THE CUSTOMER JOURNEY

## Customer Journey

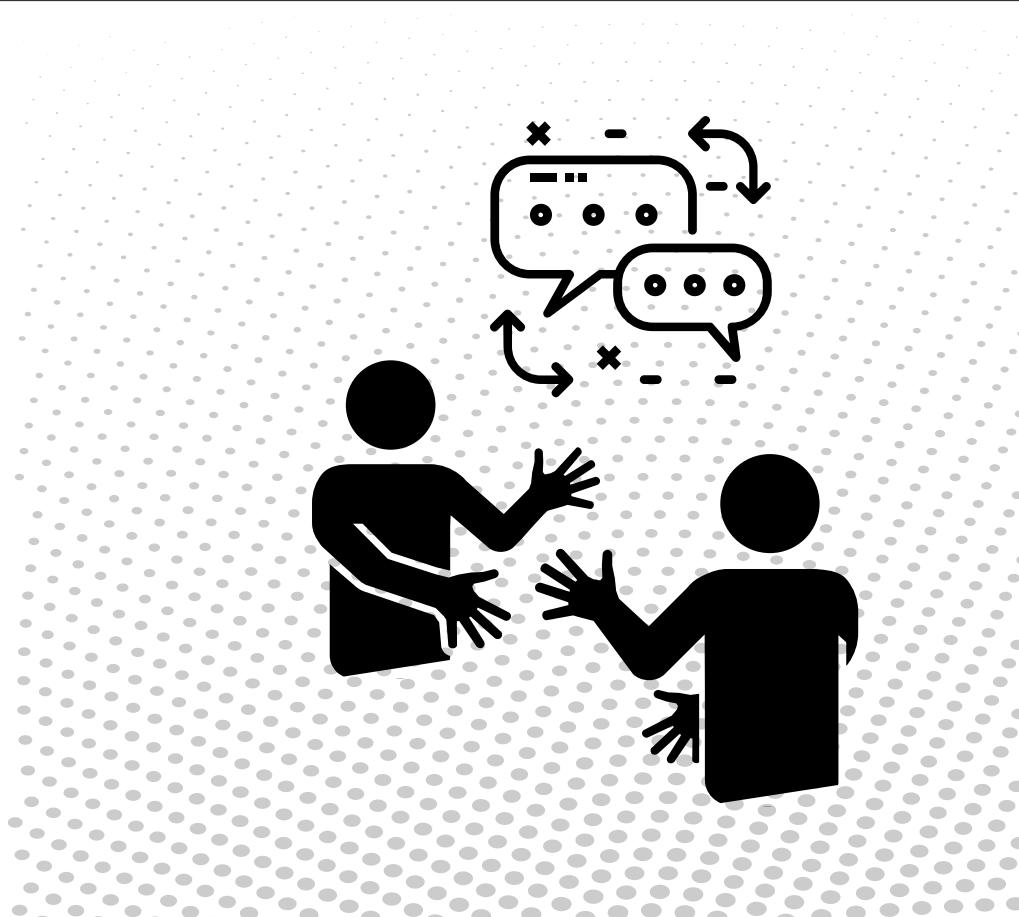


# P6: REDUCE LANGUAGE COMPLEXITY

*I came across a set of API resources for managing a DEG the other day. You could add, update, delete and get DEGs. You can also pull analytics, history, and other elements of a DEG. I spent about 10-15 minutes looking around their developer portal, documentation, and even Googling, but never could figure out what a DEG was. Nowhere in their documentation did they ever tell consumers what a DEG was, you just had to be in the know I guess.*

— Kin Lane

# [NOT] RECOGNIZING FAMILIAR TERMS



# BACKGROUND KNOWLEDGE SECTIONS

## Background Knowledge

Because Amazon's Fire OS is based on Android, Amazon tries to maintain as much parity with Android development as possible. Because of this, the documentation here doesn't duplicate the information in the Android documentation; instead, it covers how Amazon and Fire OS differs. For a better understanding of the concepts here, consult these foundational Android documentation topics:

- [Device Compatibility](#)
- [Filters on Google Play](#)
- [Supporting Different Platform Versions](#)
- [Permissions that Imply Feature Requirements](#)
- [Multiple APK Support](#)
- [Creating Multiple APKs for Different API Levels](#)

It will also help to understand some common terms.

Key terms

For more glossary definitions, see the [App Submission Glossary](#).

# P7: ITERATIVE DESIGN OF DOCS

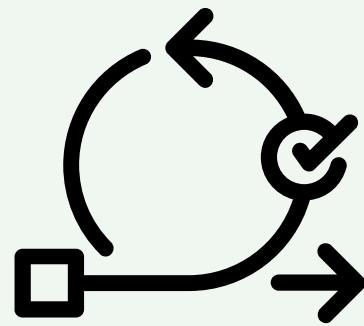
# DOCS AS A "THEORY" TO TEST

*In a sense, many things on a software development project are theories, or more accurately, assertions that need to be evaluated. ...*

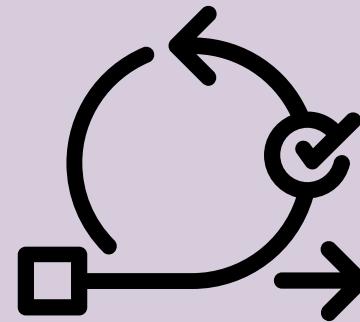
*Just because some stakeholders ... say a requirement is valid does not mean that they are correct. We need to evaluate even the requirements to determine whether they define the right solution to the problem at hand. –*

*Spence and Bitner*

# UPON INITIAL PUBLICATION, BEGIN ITERATIVE CYCLES



Software  
development



Documentation  
development

# RECAP OF ARGUMENT

- Technology is getting simpler on the front-end for end-users
- But the code underneath is becoming increasingly specialized/complex
- Tech writers are generalists, not specialists
- To provide value in specialist contexts, tech writers must exploit the gaps
- These gaps are (1) doc tools/processes, (2) understanding user feedback/experiences, and (3) information usability

# WHERE TO GET MORE INFO

The screenshot shows a website with a header featuring a yellow square icon and the text "I'd Rather Be **Writing**" with a subtitle "exploring technical writing trends and innovations". The navigation bar includes links for "About", "Archives", "Jekyll", "Academics & Practitioners", "Beginners", "Complexity", "API doc", and "Podcasts", along with a search bar. The main content area displays a blog post titled "Tech comm trends: Providing value as a generalist in a sea of specialists (Part I)". The post is categorized under "Series" and "Part I: Introduction and argument overview". It was written by Tom Johnson on Oct 2, 2018, and includes categories like "api-doc", "simplifying-complexity", and "writing". A summary discusses the shift from end-user to developer domains and the role of generalists. On the left sidebar, there's a newsletter sign-up form, social media links (RSS, Twitter, LinkedIn, etc.), and a "Recent Comments" section with a comment from Eric Weston.

Essay series: <http://bit.ly/genandspecialisttrendspart1>

# INTERACTIVE SURVEYS

## Your reactions and input

UX professionals have reduced the need for technical writers to provide documentation for mainstream end-users.

- Strongly Disagree
- Disagree
- Undecided
- Agree
- Strongly Agree

Many technical writing jobs have shifted from the end-user domain to the developer domain.

# QUESTIONS?



# THE END



*Tom Johnson*

- [idratherbewriting.com](http://idratherbewriting.com)
- [@tomjohnson](https://twitter.com/tomjohnson)
- [tomjoht@gmail.com](mailto:tomjoht@gmail.com)