
# Feature and Capability Breakdown; Sizing; Thin-Slicing

Jon Kolko
Professor, Austin Center for Design

ac4d

# Feature and Capability Breakdown

A process of creating individual, atomic pieces of functionality for product development

# Feature and Capability Breakdown

## A process of creating individual, atomic pieces of functionality for product development

BREAKING DOWN CAPABILITIES...

1. Moves a design from “blue sky” to “realistic”
2. Helps you think about reusable components, leading to efficiencies
3. Gets ready for sizing

ac4d

# Product vs. Design: Shipping

Design is about visualizing an optimistic future, and painting a picture of *what if*.

Shipping is about reality – actually bringing a product to market so people can use it and benefit from it.

| Design | Product Management |
| --- | --- |
| Long-term view of an optimistic future | Long-term view of an optimistic future |
| A desire to have people experience *the thing* | A desire to have people experience *the thing* |
| The luxury of unlimited resources | Always resource constrained |
| Focused on the ideal | Focused on the pragmatic |

You are both!

ac4d

# Temporarily abandoning our user stance

We focus on people, their goals, and the way the system will support their experience in achieving those goals.

A focus on features, components, and controls explicitly rejects the idea of “user centered”, and instead focuses on “technology centered.”

Why would we do this?

# How developers think

Developers…

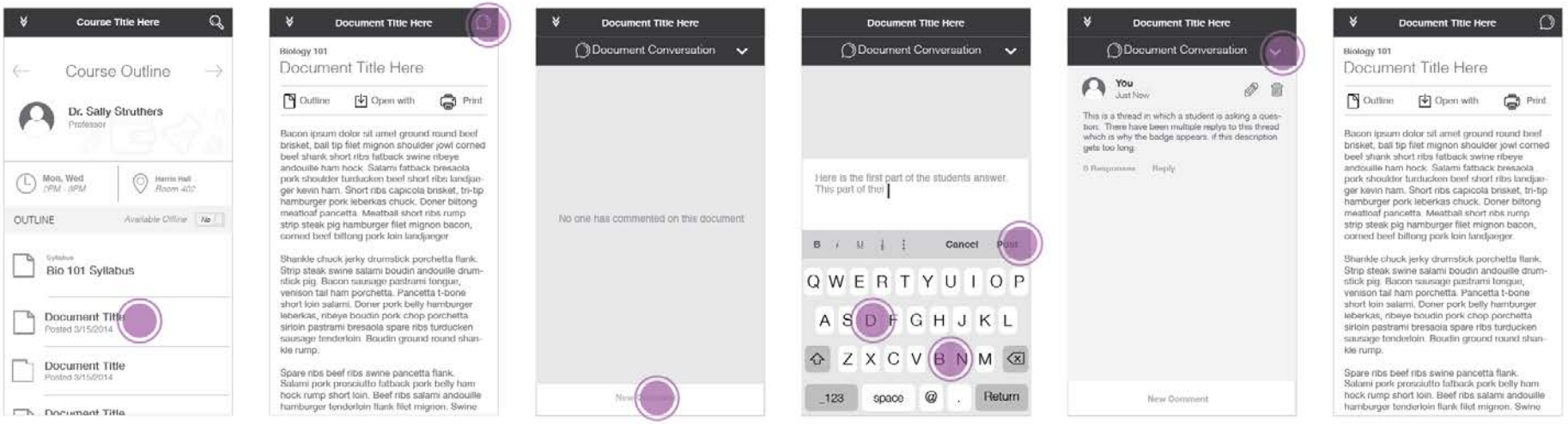
- Strive for efficiency
- Aim for optimization
- Hate doing the same work twice
- Value their time
- Tend towards “well-defined” problems

Why is this important, at this stage in the process?

# Start with an end-to-end flow

To identify features and capabilities, you need detailed, comprehensive wireframes of the entire “ideal” flow (the hero path). Visualize these in a single canvas, in sequence, and then print them – large – and put them on the wall.

*Open a document and hold a discussion about it*



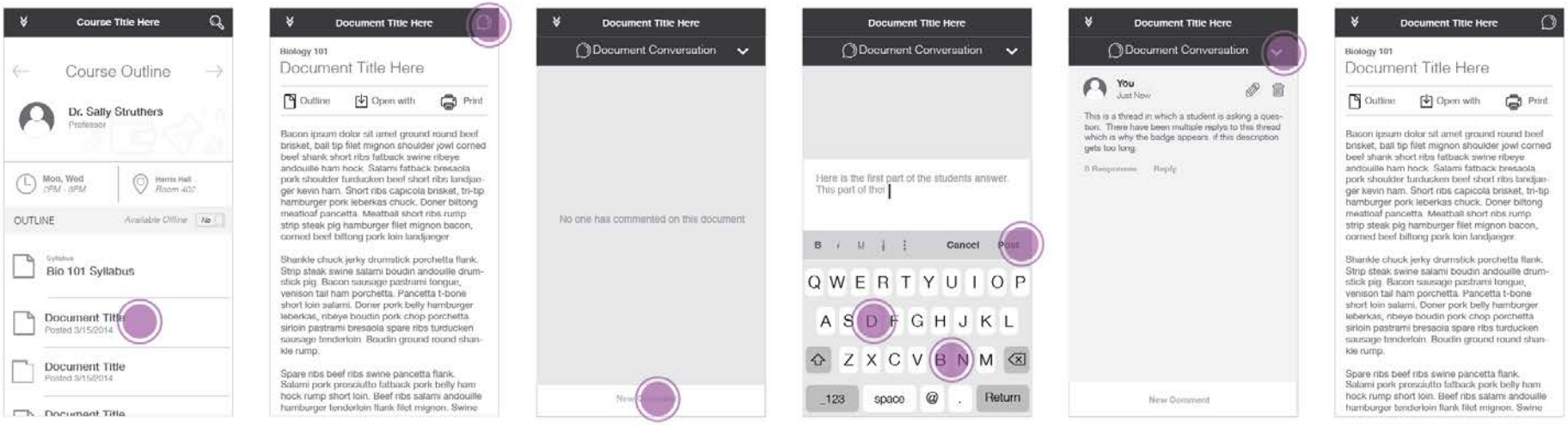

ac4d

# Identifying unique features

Walk through each screen, and identify each unique feature.

## What is a feature?

*Open a document and hold a discussion about it*

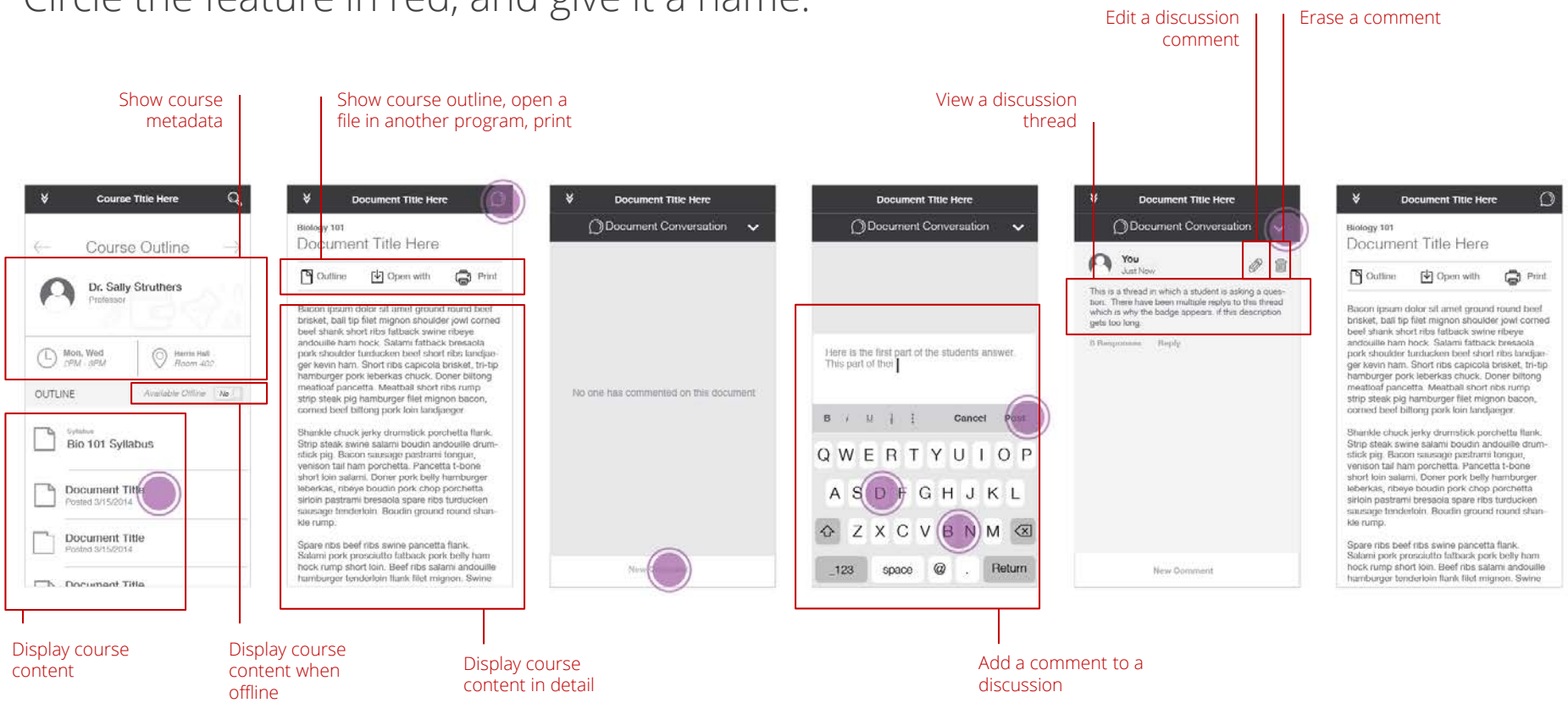



ac4d

# Identifying unique features

Walk through each screen, and identify each unique feature.

Circle the feature in red, and give it a name.




ac4d

# Identifying unique components & controls

Walk through each screen, and identify each unique component or control.

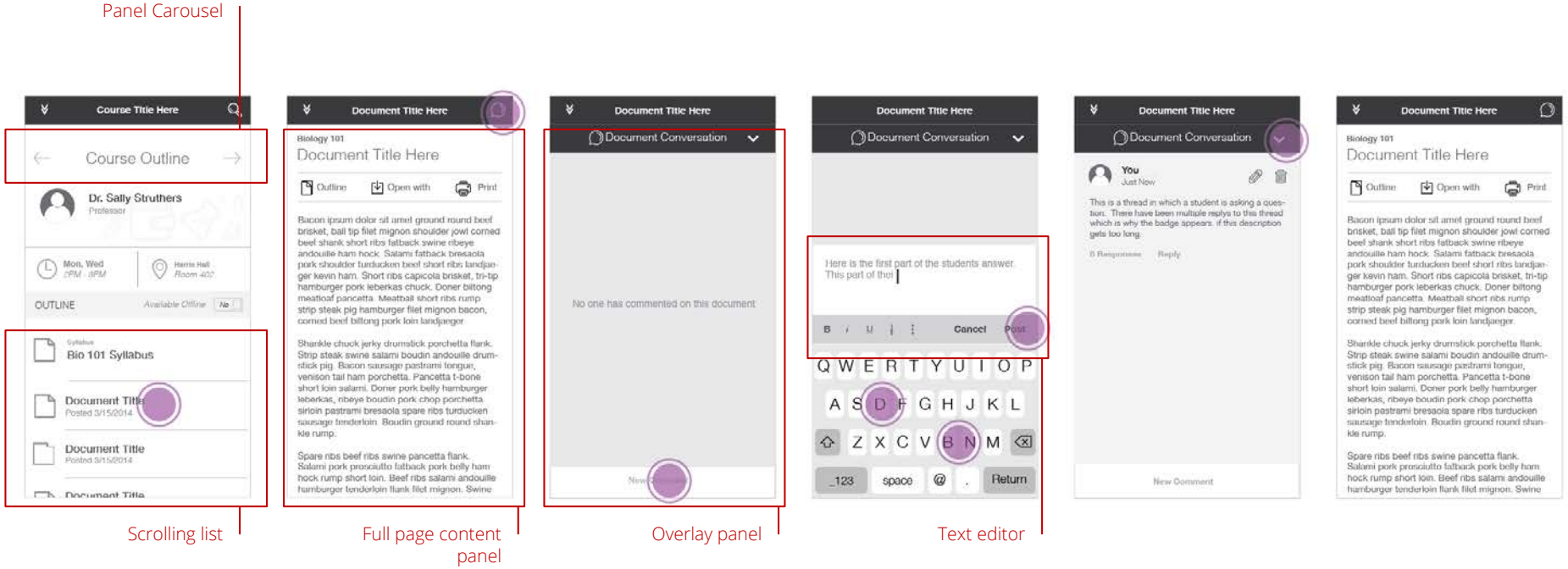
What is a component or control?

ac4d

# Identifying unique components & controls

Walk through each screen, and identify each unique component or control.

Circle the component or control in red, and give it a name.

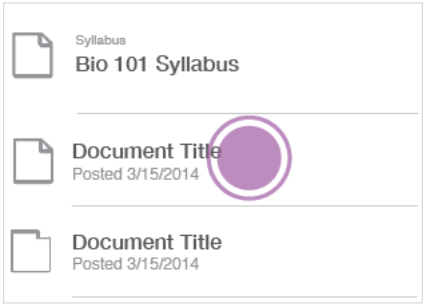



ac4d

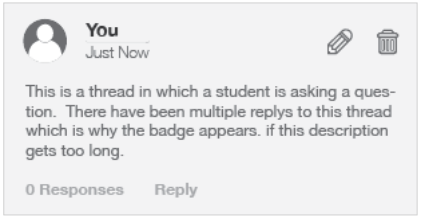
# Documenting the system

Create a single representation of each feature, component, or control, outside of the context of the wireframes.

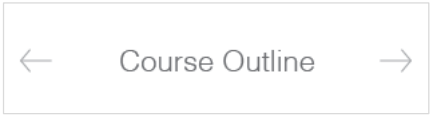
- Label them with a name
- Give them a unique identifier
- Write a description under each element
- Provide a reference to example wireframes and flows, to show the element in context




1.0
Display Course Content
The ability to present an icon, a document title, and the date it was posted




2.0
View a Discussion Thread
The ability to present a discussion response, a user avatar, a user name, the post date, and the number of responses




Component A.0
Panel Carousel
The ability to display panels and navigate through them by swiping left or tight

ac4d

# Feature and Capability Breakdown

## A process of creating individual, atomic pieces of functionality for product development

BREAKING DOWN CAPABILITIES...

1. Moves a design from “blue sky” to “realistic”
2. Helps you think about reusable components, leading to efficiencies
3. Gets ready for sizing

HOW TO DO IT

1. Start with a hero flow
2. Identify unique features
3. Identify unique components & controls
4. Document the system

ac4d

# Sizing

A method for assigning relative estimates to unique features, components, and controls

# Sizing

## A method for assigning relative estimates to unique features, components, and controls

SIZING...

1. Helps identify total time on task for a development effort
2. Helps steer redesign efforts to launch product quickly
3. Helps guide prioritization activities
4. Is often dramatically inaccurate

ac4d

# What are we talking about?

Software development has a number of models (this is an abstraction, and not a hard-fast rule):

1. Small group (Startup).
   Whiteboard some stuff, code it, ship it.

2. Small group (Agile).
   Identify a product backlog of capabilities, prioritize them in small “sprints” based on importance and size, ship them.

3. Large group (Waterfall).
   Design the whole system, begin development, work till it’s done (typically towards specific dates and milestones), ship it.

# Reasons to size

Sizing before (and during) development:

- Provides clarity to stakeholders who have to commit to delivery (to a board of directory, for example)
- Provides clarity to marketers, who need to build campaigns around delivery dates
- To understand an order of magnitude for producing a product
- It gives the entire team a view of the product, so they understand it thoroughly
- It forces detailed interactions between the designers and developers

# Reasons not to size

Sizing before (and during) development:

- It's extraordinarily inaccurate
- It's extraordinarily inaccurate
- It's extraordinarily inaccurate

ac4d

# Understanding developer “man days”

Development estimation is considered in business days, often called man days (how sexist!)

## Capacity

- One developer working for a business day = one man day.
- A team of five developers working for a business day = five man days.
- A team’s available capacity is considered the total of their days to work per week

ac4d

# Understanding developer “man days”

## Estimates

A team of 5 working for 4 weeks (20 business days) has 100 total days capacity. But, they only have 25 days capacity per week. If a feature will take 80 total man days, the feature will get done in approximately 3 weeks.

“Can’t we just add more people, and get it done faster?”

ac4d

# T-Shirt Sizing, Fibonacci Sizing

Two different sizing mechanisms are typically used:

- T-Shirt Sizing

  Small (1 day)
  Medium (10 days)
  Large (20 days)
  X-Large (40 days)

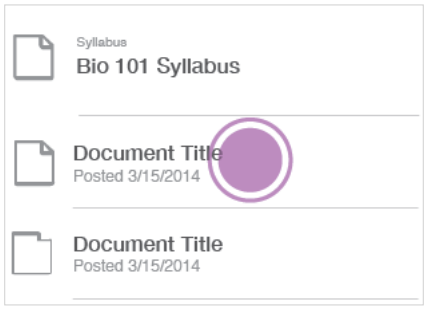
- Fibonacci Sizing

  0, ½, 1, 2, 3, 5, 8, 13, 20, 40, 100

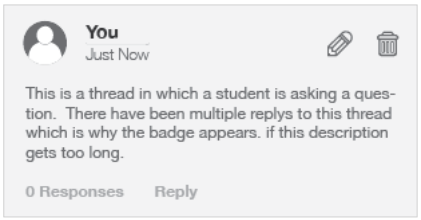
# Facilitating an estimation session

To facilitate a sizing/estimation session:

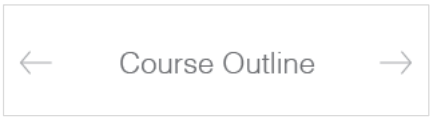
1. Print the features, components, and controls – large.
2. Walk through each item. Describe it, and ask if there are questions.
3. The developers all privately assign the item an estimate
4. Then, average the estimates across all developers
5. Write the estimate both on the capability directly, and in a spreadsheet




1.0
Display Course Content
The ability to present an icon, a document title, and the date it was posted




2.0
View a Discussion Thread
The ability to present a discussion response, a user avatar, a user name, the post date, and the number of responses




Component A.0
Panel Carousel
The ability to display panels and navigate through them by swiping left or tight

ac4d

# Documentation

| Screen ID | Screen Title | Flow found on.. | Estimate | Discussion |
|---|---|---|---|---|
| 1.0 | Display Course Content | Create a discussion | 14 days | There's confusion around where the data would be saved. May take longer. |
| 2.0 | View a Discussion Thread | Create a discussion | 7 days | |
| A.0 | Panel Carousel Component | Create a discussion | 30 days | Reusable component |

ac4d

# Sizing

## A method for assigning relative estimates to unique features, components, and controls

SIZING...

1. Helps identify total time on task for a development effort
2. Helps steer redesign efforts to launch product quickly
3. Helps guide prioritization activities
4. Is often dramatically inaccurate

HOW TO DO IT

1. Start with your feature, component and control output
2. Conduct a facilitation session
3. Walk through each element
4. Answer questions and clarifications about the design
5. Assign a relative t-shirt size
6. Recalibrate early estimates based on later estimates
7. Document the sizing in a spreadsheet

ac4d

# Thin-Slice Hero Flow

Minimizing a hero-flow so it fits into a development timeline yet retains user value

# Thin-Slice Hero Flow

## Minimizing a hero-flow so it fits into a development timeline yet retains user value

THIN SLICING...

1. Balances realism of development capacity with a vision to support users
2. Continues to ground product in flows, rather than features

ac4d

# Identify capacity & compare to estimates

Identify your *total* capacity in mandays

Compare your capacity to your estimates to get a gauge for how much of a disconnect you have

ac4d

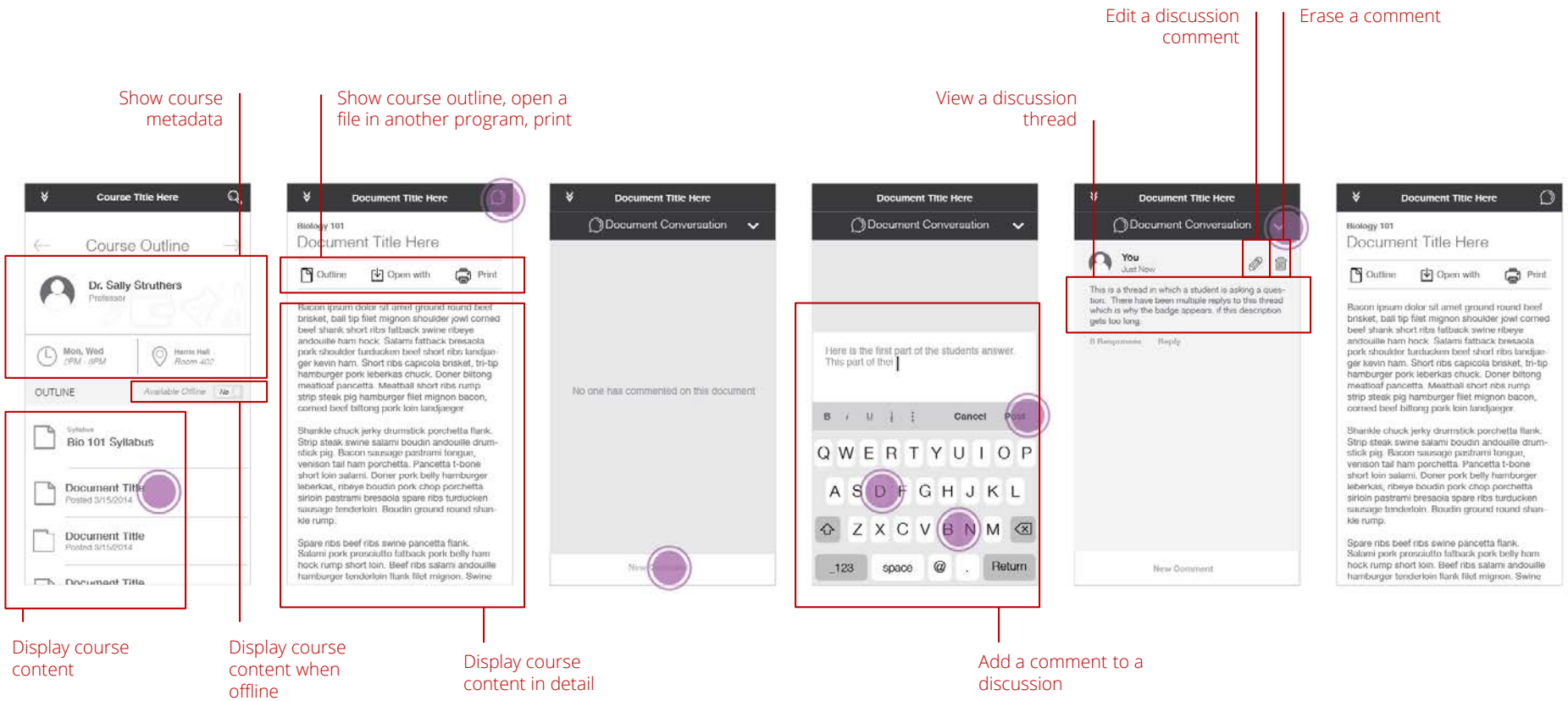
# Create “rough cut” thin slice flows

Working flow-by-flow, remove things:

- So that the capabilities fit into your capacity, AND
- So that the flows still make sense, AND
- So that the flows still offer value to a user

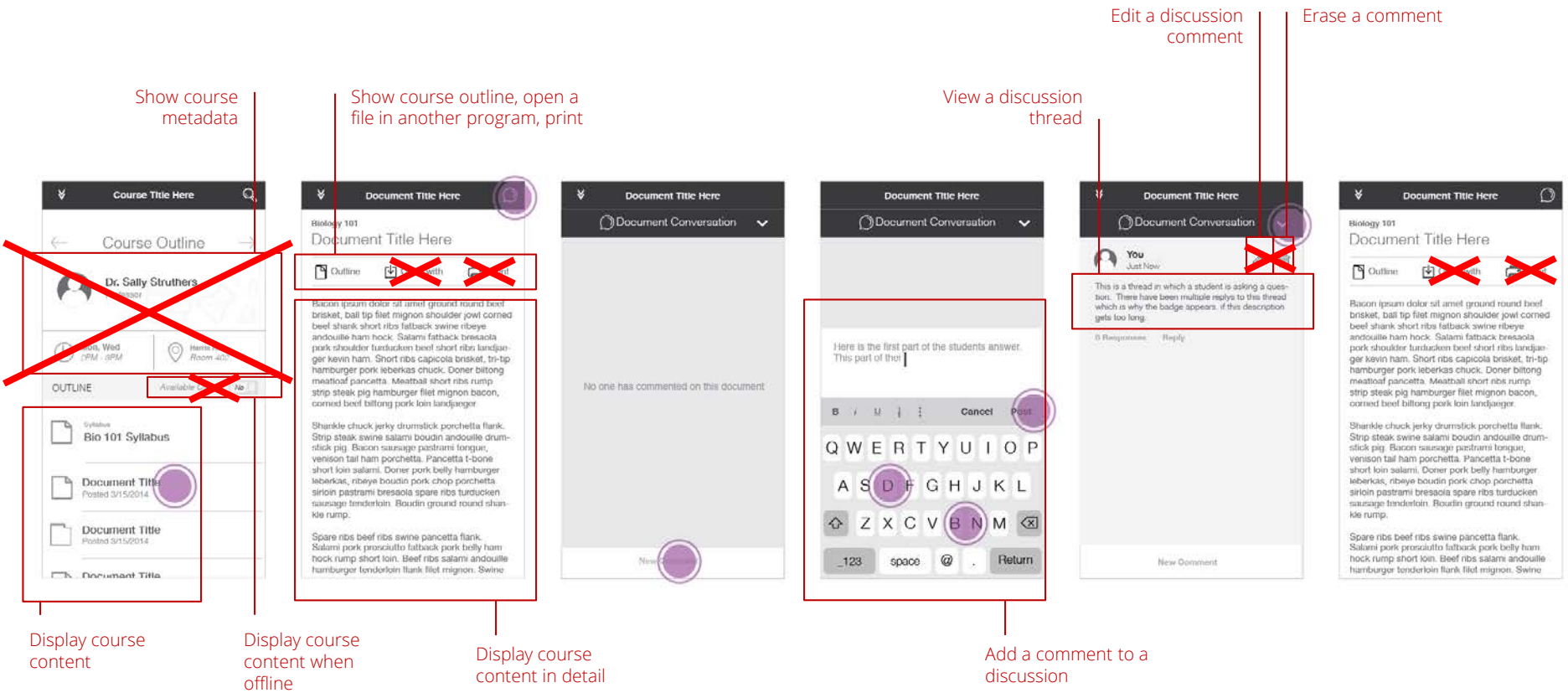
ac4d

# Create “rough cut” thin slice flows




ac4d

# Create “rough cut” thin slice flows




ac4d

# Rationalize with sizing estimates

Resize the hero flow, and see if it now fits into your capacity numbers. If not... cut more.

ac4d

# Thin-Slice Hero Flow

## Minimizing a hero-flow so it fits into a development timeline yet retains user value

THIN SLICING...

1. Balances realism of development capacity with a vision to support users
2. Continues to ground product in flows, rather than features

HOW TO DO IT

1. Identify total capacity based on available resources
2. Create “rough-cut” view of thin slice, and visualize wires as a hero flow
3. Rationalize with sizing estimates, comparing the features that remain with the overall estimates
4. Refine the flows if they are under or over capacity

ac4d

# ac4d

Jon Kolko
Director, Austin Center for Design
jkolko@ac4d.com