Module 1

Introduction to Data Visualization

- ❑ Overview of data visualization
- ❑ Data Abstraction
- ❑ **Task Abstraction**
- ❑ **Analysis: Four Levels for Validation**

Text Book

Tamara Munzer**, Visualization Analysis and Design** -, CRC Press 2014 . **(Chapter 1, 2,3 and 4)**

Task Abstraction

Outline

- Why Analyze Tasks Abstractly?
- Who: Designer or User
- Actions
  - Analyze
    - Consume, Produce
  - Search
    - Lookup, Browse, Locate, Explore
  - Query

Why Analyze Tasks Abstractly?

- Vis framework encourages you to consider tasks in ***abstract*** form, rather than the ***domain-specific*** way.
- Transforming task descriptions from domain-specific language into abstract form allows you to reason about ***similarities*** and ***differences*** between them.
- Otherwise, it's ***hard to make useful comparisons*** b/w domain situations, because if you don't do this kind of translation then everything just appears to be different. That apparent difference is ***misleading***.

Who: Designer or User

- Who has a goal or makes a design choice: the designer of the vis or the end user of the vis.
- The designer has built many choices.
- Tools are limited(kinds of data and tasks that they can address), but their strength is that users are not faced with an overwhelming array of design choices.
- The breadth of choices is both a strength and a limitation:

Who: Designer or User

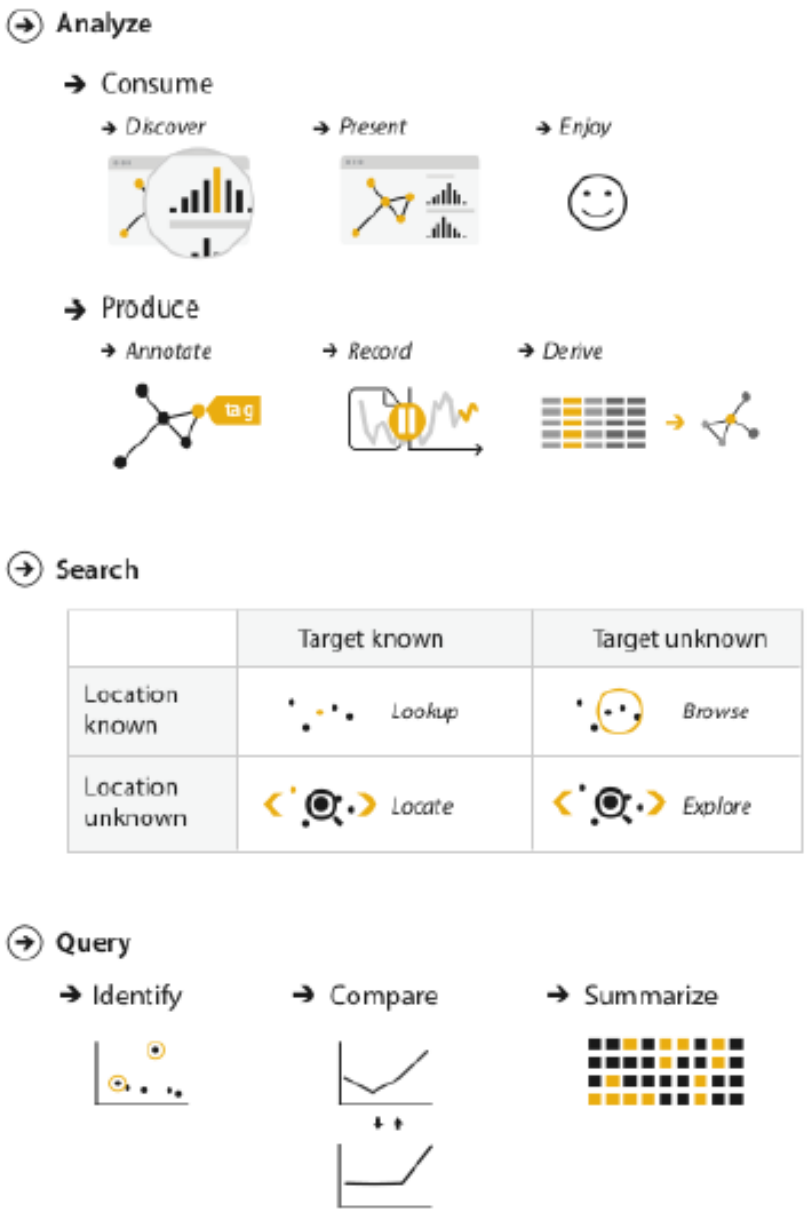
- Users have a lot of power, but they also may make ineffective choices if they do not have a deep understanding of many vis design issues.

Actions

- Three levels of actions that define user goals.
- The high-level choices describe how the vis is being used to analyze, either to consume existing data or to also produce additional data.
- The mid-level choices cover what kind of search is involved, in terms of whether the target and location are known or not.
- The low-level choices pertain to the kind of query: does the user need to identify one target, compare some targets, or summarize all of the targets?

Actions

The choices at each of these ***three levels*** are independent from each other, and it's usually useful to describe actions at all three of them.



Actions - Analyze

- Two possible goals

  1.Consume existing information(common).

  - Discover something new
  - Analyze information that is not already completely understood
  - For users to enjoy a vis to indulge their casual interests in a topic

➔ Consume

➔ *Discover* ➔ *Present* ➔ *Enjoy*

Actions - Analyze

- Two possible goals

  2.Actively produce new information.
- There are three kinds of produce goals:
  - Annotate
  - Record
  - Derive

Search

- All of the high-level analyze cases require the user to ***search*** for elements of interest within the vis as a mid-level goal.
- The classification of search into four alternatives according to search target is ***already known or not***.

	Target known	Target unknown
Location known	 *Lookup*	 *Browse*
Location unknown	 *Locate*	 *Explore*

Search - Lookup

- If users already know ***both*** what they're looking for and where it is, then the search type is simply lookup.

- Example, a user of a tree vis showing the ancestral relationships between mammal species might want to look up humans, and can get to the right spot quickly by remembering how humans are classified.

Search - Locate

- To find a known target at an unknown location, the search type is locate: that is, find out where the specific object is.

- User might not know where to find rabbits, and would have to look around in a number of places before locating them as lagomorphs (not rodents)!

Search - Browse

- Users don’t know exactly what they’re looking for, but they do have a location in mind of where to look for it, the search type is browse.
- For instance, if a user of a tree vis is searching within a particular subtree for leaf nodes having few siblings, it would be an instance of browse because the location is known in advance, even though the exact identity of the search target isn’t.

Search - Browse

- When users are not even sure of the location, the search type is explore. It entails searching for characteristics without regard to their location, often beginning from an overview of everything.
- Examples - Searching for outliers in a scatterplot, for anomalous spikes or periodic patterns in a line graph of time-series data.

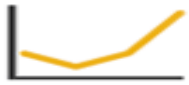
Query

- Once a target or set of targets for a search has been found, a low-level user goal is to query these targets at one of three scopes:
  - Identify - refers to a single target
  - Compare - refers to multiple targets
  - Summarize - refers to the full set of possible targets.

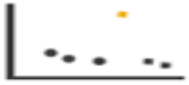
Targets

All Data

- Trends



- Outliers



- Features

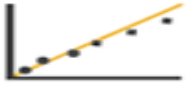
Attributes

- One
  - *Distribution*

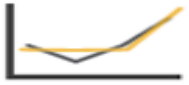
  - *Extremes*

- Many
  - *Dependency*

  - *Correlation*



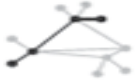
  - *Similarity*



Network Data

- Topology

- *Paths*



Spatial Data

- Shape

How?

Encode

- Arrange
  - Express
  - Separate
  - Order
  - Align
  - Use

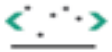
- Map
  from categorical and ordered attributes
  - Color
    - *Hue*
    - *Saturation*
    - *Luminance*
  - Size, Angle, Curvature, ...
  - Shape
  - Motion
    Direction, Rate, Frequency, ...

Manipulate

- Change

- Select

- Navigate

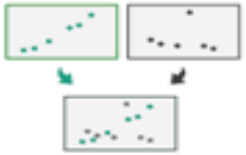


Facet

- Juxtapose

- Partition

- Superimpose



Reduce

- Filter

- Aggregate

- Embed

Analysis: Four Levels for Validation

Outline

- Why Validate?
- Four Levels of Design
- Angles of Attack
- Threats to Validity
- Validation Approaches
- Validation Examples

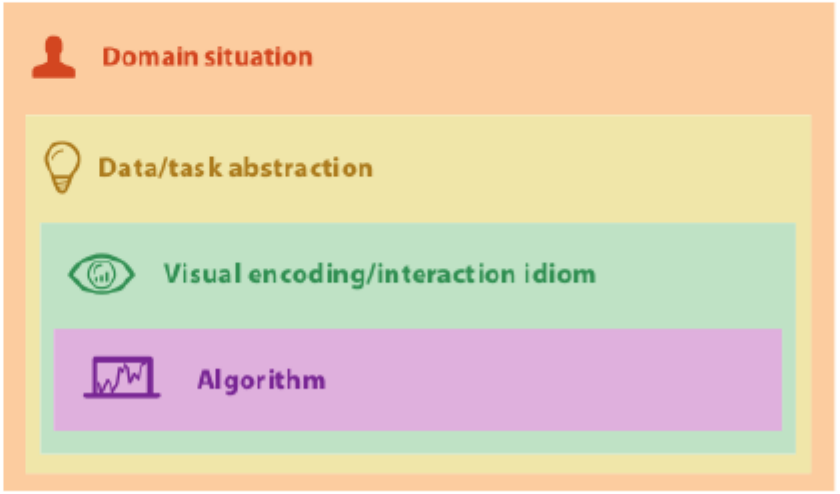
Why Validate?

- Most designs are ineffective and validation is a tricky problem that is difficult to get right.

- It’s valuable to think about how you might validate your choices from the very beginning of the design process, rather than leaving these considerations for the end as an afterthought.

Four Levels of Design

- Splitting the complex problem of vis design into four cascading levels provides an analysis framework that lets you address different concerns separately.

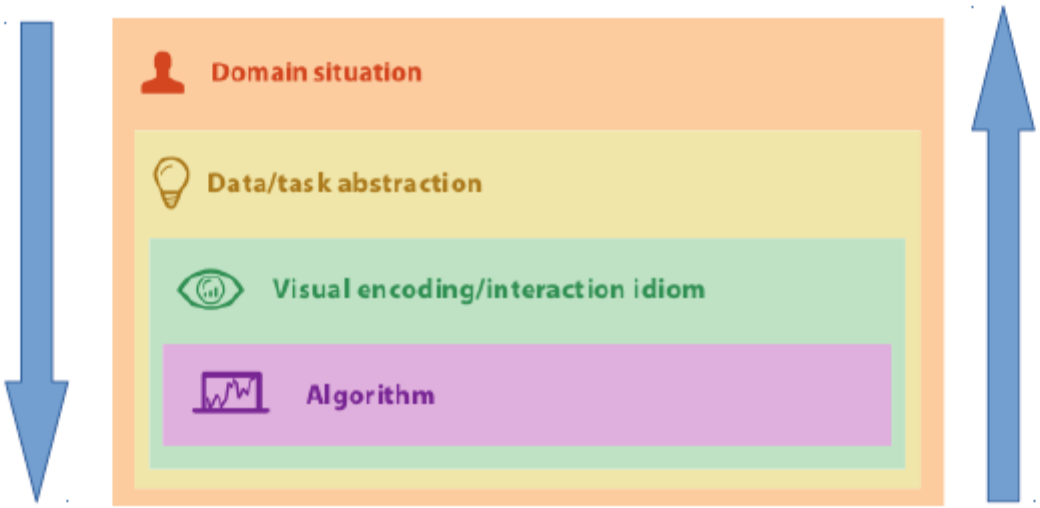


Four Levels of Design

- At the top is the situation level, where you consider the details of a particular *application domain* for vis.
- Next is the *what–why abstraction level*, where you map those domain-specific problems and data into forms that are *independent of the domain*.
- The following how level is the design of idioms that specify the approach to *visual encoding* and *interaction*.
- Finally, the last level is the *design of algorithms* to instantiate those idioms computationally

Angles of Attack

- There are two common angles of attack for vis design: top down or bottom up.



top down (problem-driven) bottom up (technique-driven)

- Considering the four levels of nested model explicitly can help you avoid the pitfall of skipping important steps

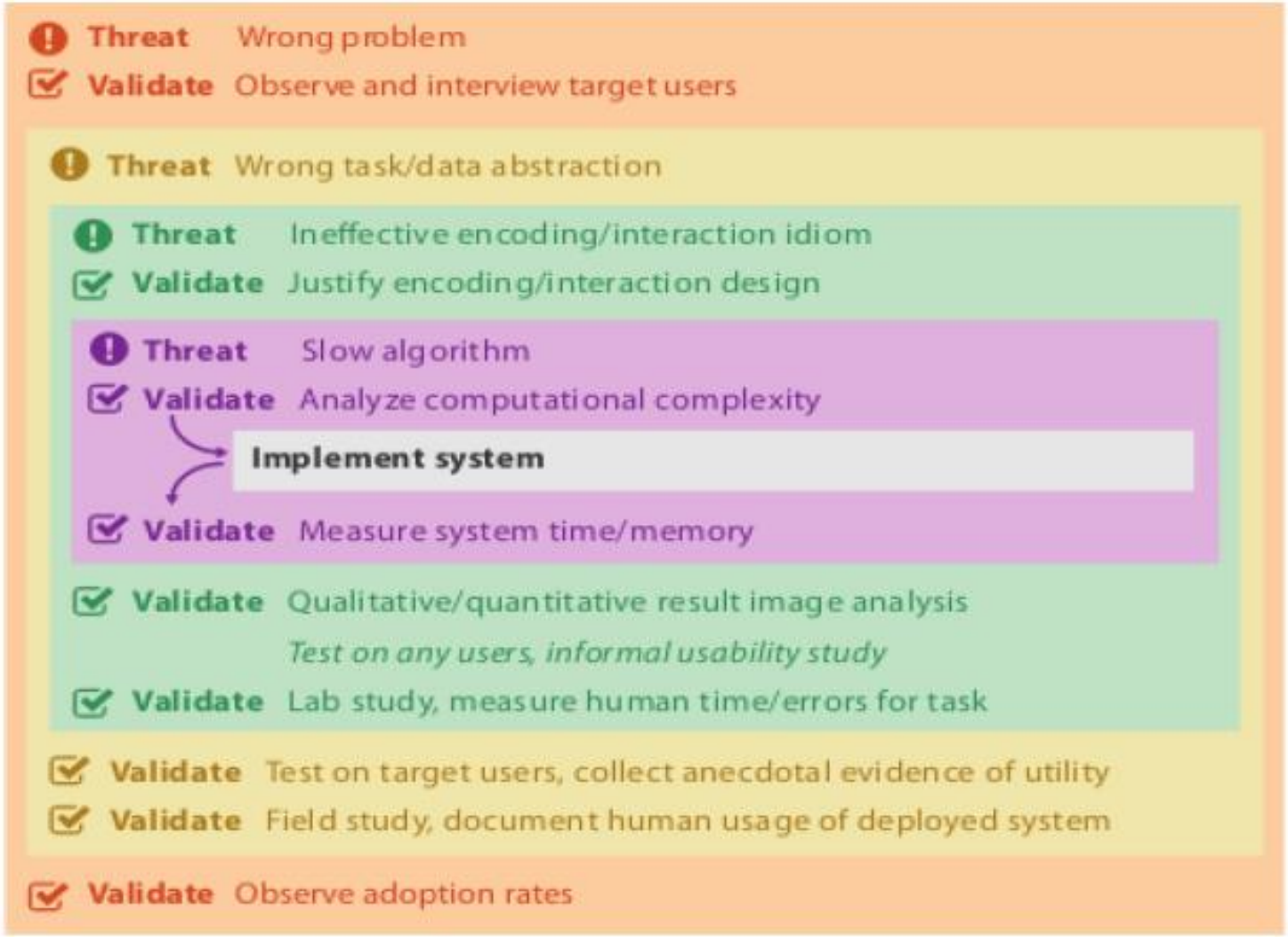
Threats to Validity

- Each of the four levels has a different set of threats to validity: that is, different fundamental reasons why you might have made the ***wrong choices***.

- Wrong problem: You misunderstood their needs.
- Wrong abstraction: You're showing them the wrong.
- Wrong idiom: The way you show it doesn't work.
- Wrong algorithm: Your code is too slow.

Validation Approaches

- Different threats require very different approaches



Validation Approaches

- Immediate Versus down-stream validation.
- Having nested levels is that most kinds of validation for the outer levels are not immediate because they require results from the downstream levels nested within them.
- Downstream dependencies add to the difficulty of validation: a poor showing of a test may misdirect attention upstream, when in fact the problem results from a poor choice at the current level.

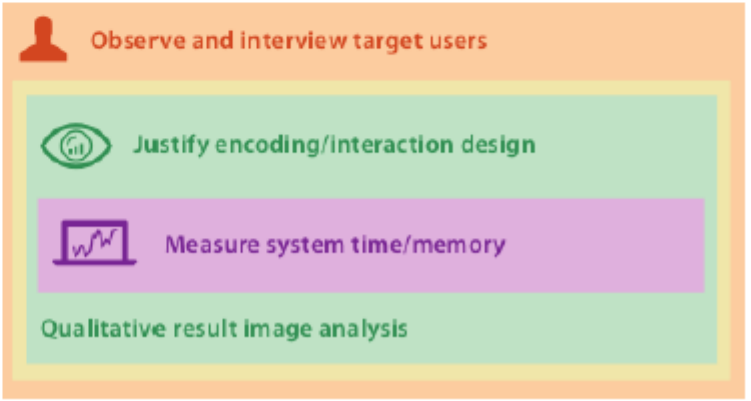
Validation Approaches

- For example, a poor visual encoding choice may cast doubt when testing a legitimate abstraction choice, or poor algorithm design may cast doubt when testing an interaction technique.
- Despite their difficulties, the downstream validations are necessary. The immediate validations only offer partial evidence of success;

Validation Examples

- Social Network Analysis- *Matrix Explorer system for social network analysis [Henry and Fekete 06],*

  At the *domain situation level*, there is explicit characterization of the *social network analysis domain*, which is validated with the *qualitative techniques of interviews and an exploratory study using participatory design methods with social scientists and other researchers* who use social network data.



Validation Examples

- At the abstraction level, the detailed list of requirements of the target user needs discussed in terms of abstract tasks and data.
- There is a thorough discussion of the primary encoding idiom design decision to use both *node–link* and *matrix views* to show the data, and also of many secondary encoding issues.
- There is also a discussion of both basic interaction idioms and more complex interaction via *interactive reordering* and *clustering*.

Validation Examples

- In both cases the authors use the *immediate validation* method of justifying these design decisions.
- There is also an extensive *downstream validation* of this level using qualitative discussion of result images.
- At the *algorithm level*, the focus is on the *reordering algorithm*. Downstream benchmark timings are mentioned very briefly