Model for visualization design with four layers: characterize the task and data in the vocabulary of the problem domain, abstract into operations and data types, design visual encoding and interaction techniques, and create algorithms to execute techniques efficiently. Output from a level above is input to the level below, bringing attention to the design challenge that an upstream error inevitably cascades to all downstream levels.

**Domain Problem and Data Characterization** 🡪 Each domain (microbiology or high-energy physics), usually has its own vocab for describing its data and problems and there is usaly an existing workflow how the data is used. This means a gap between designers and users.

**Operation and Data Type Abstraction 🡪** The abstraction stage is to map problems and data from the vocabulary of the specific domain into a more abstract and generic description that is in the vocabulary of info visualization: output of this level is a description of operations (generic rather than domain-specific tasks) and data types 🡪 forms input required for making visual encoding decisions at the next level.  
 The other aspect of this stage is to transform the raw data into the data types that visualization techniques can address.

**Visual encoding and interaction design 🡪** We consider visual encoding and interaction together rather than separately because they are mutually interdependent.

**Algorithm design 🡪** The innermost level is to create an algorithm to carry out the visual encoding and interaction designs automatically.

**Threats and Validation 🡪** Validation entails whether one has built the right product and whether one has built the product right.

**Iterative Loops and Rapid Prototyping 🡪** Although this model is cast as four nested layers for simplicity, in practice these four stages are rarely carried out in strict temporal sense.

**Domain threats** 🡪Wrong problem.  
Validate by 🡪

* observe and interview target users
* observe observation rates.

**Abstraction threats 🡪** Bad data/operation abstraction (chosen operations and data types do not solve the characterized problems of the target audience.  
Validate by -->

* Test on target users, collect anecdotal evidence of utility.
* Field study, document human usage of deployed system.

**Encoding and Interaction threats 🡪** Chosen design is not effective at communicating the desired abstraction to the user.  
Validate by 🡪

* Qualitative/quantitative result image analysis (test on any users, informal usability study).
* Lab study, measure human time/errors for operation.

**Algorithm Threats 🡪** Slow or incorrect algorithm.  
Validate by 🡪

* Analyze computational complexity.
* Measure system time/memory.

