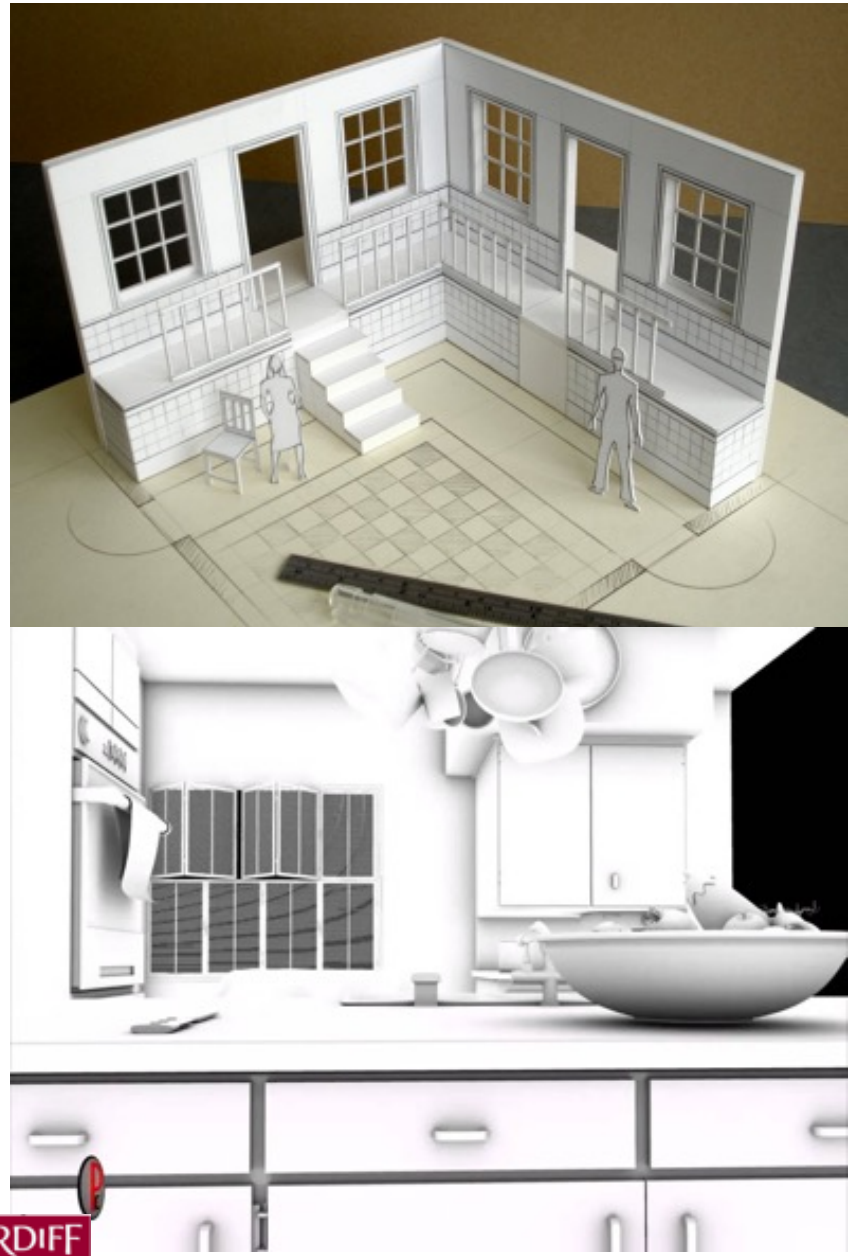
APPLICATION AND SOLUTION DOMAIN

- **Requirement Analysis - Application Domain**
  - The environment in which the system is or will be working
    - Requirement Elicitation e.g., qualitative and quantitative user studies
    - Requirement Analysis (RA) e.g., scenarios, user stories

- **Solution/System Domain**
  - **How to model?**
    - Context & Function modeling -> linked to RA
    - Static (Structure) modeling
    - Dynamic (Behavior) modeling

MODELING



https://davidneat.wordpress.com/methods/white-card-models-for-filmtv-work/

http://www.najwakitchen.com/kitchen-modeling/

http://www.najwakitchen.com/wp-content/uploads/2016/03/kitchen-modeling-images11.jpg

CARDIFF UNIVERSITY
PRIFYSGOL CAERDYDD
VerdezotoDiasN@Cardiff.ac.uk

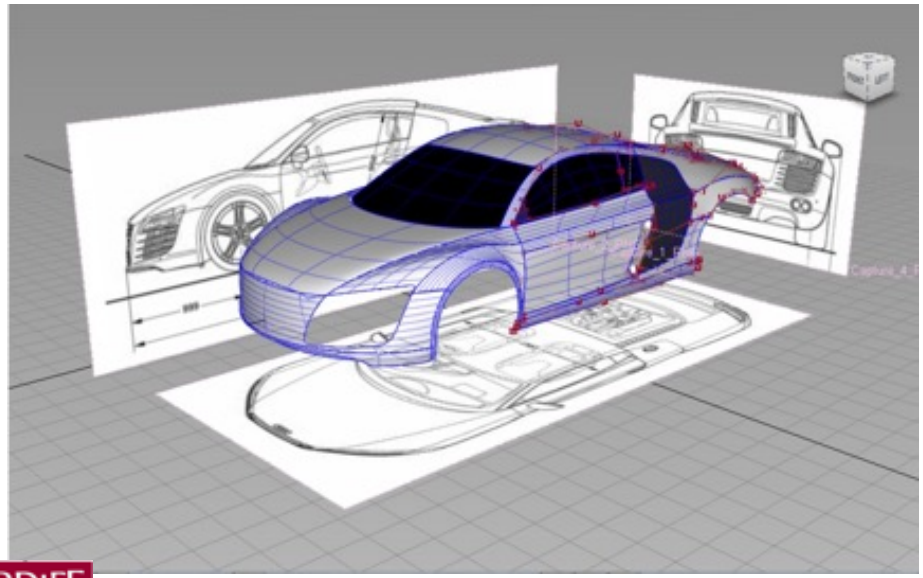
MODELING

- Builds an **abstraction** of reality
- Make a visual representation of something
  - Irrelevant details vs. relevant details
  - **Reduce complexity**
- The **degree of relevance** depends on the purpose of the model

MODELING

http://webneel.com/maya-3d-models

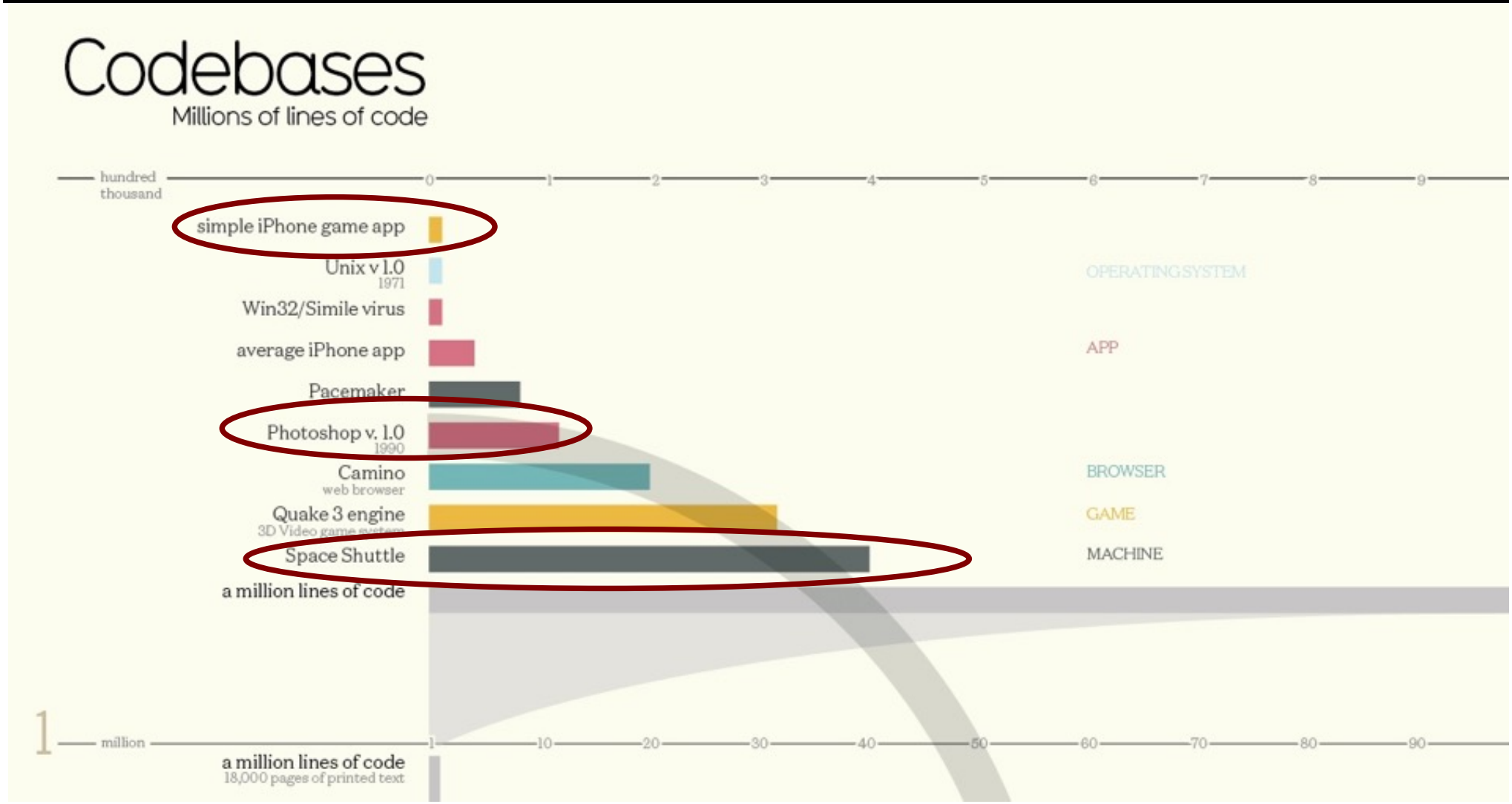
https://www.pinterest.com/pin/537758011727821842/



mlautomotivedesign.weebly.com/alias-automotive-3d-cad.html

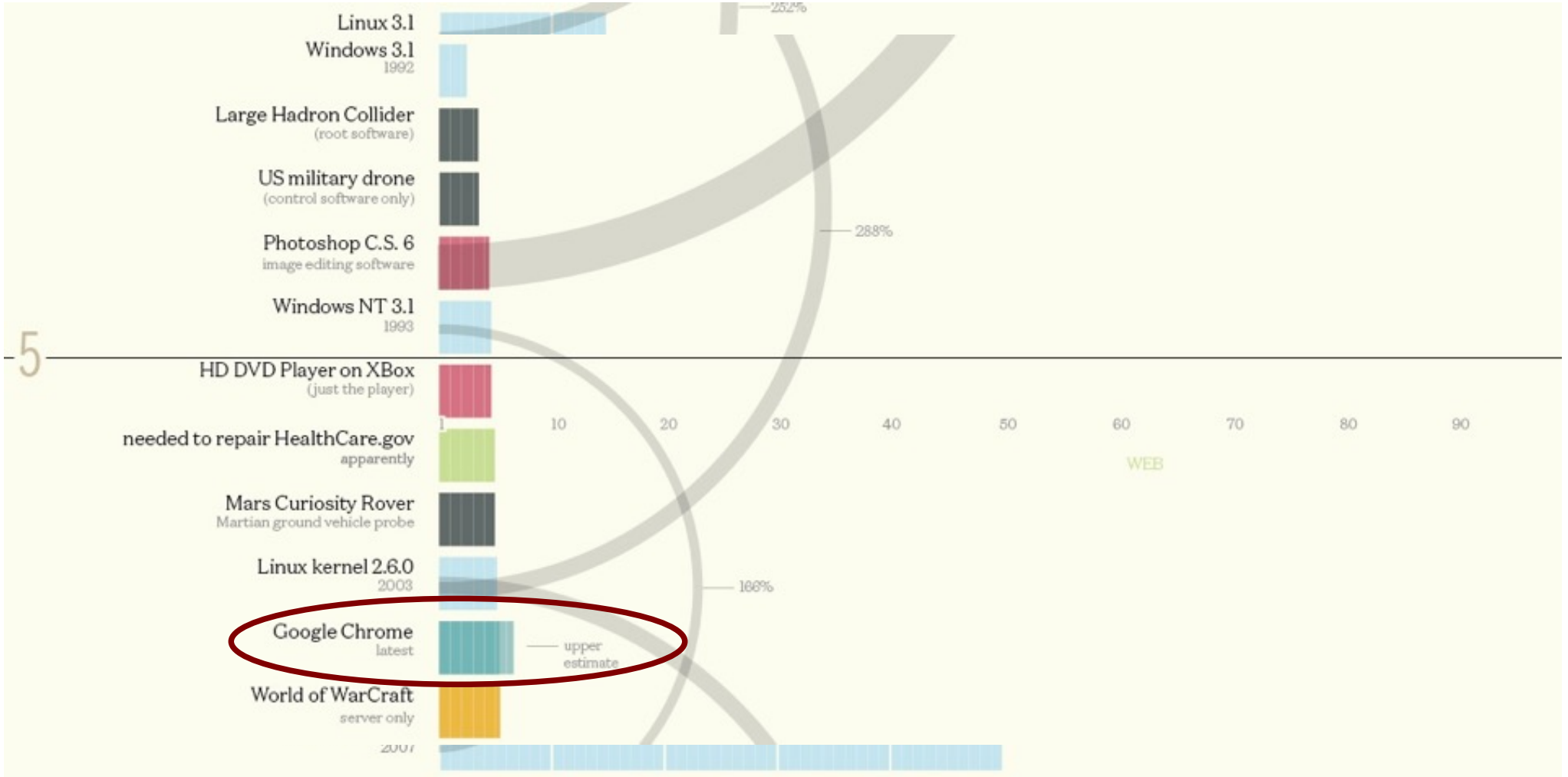
http://www.autodesk.com/products/alias-products/features/all/gallery-view

WHY MODELING SOFTWARE?

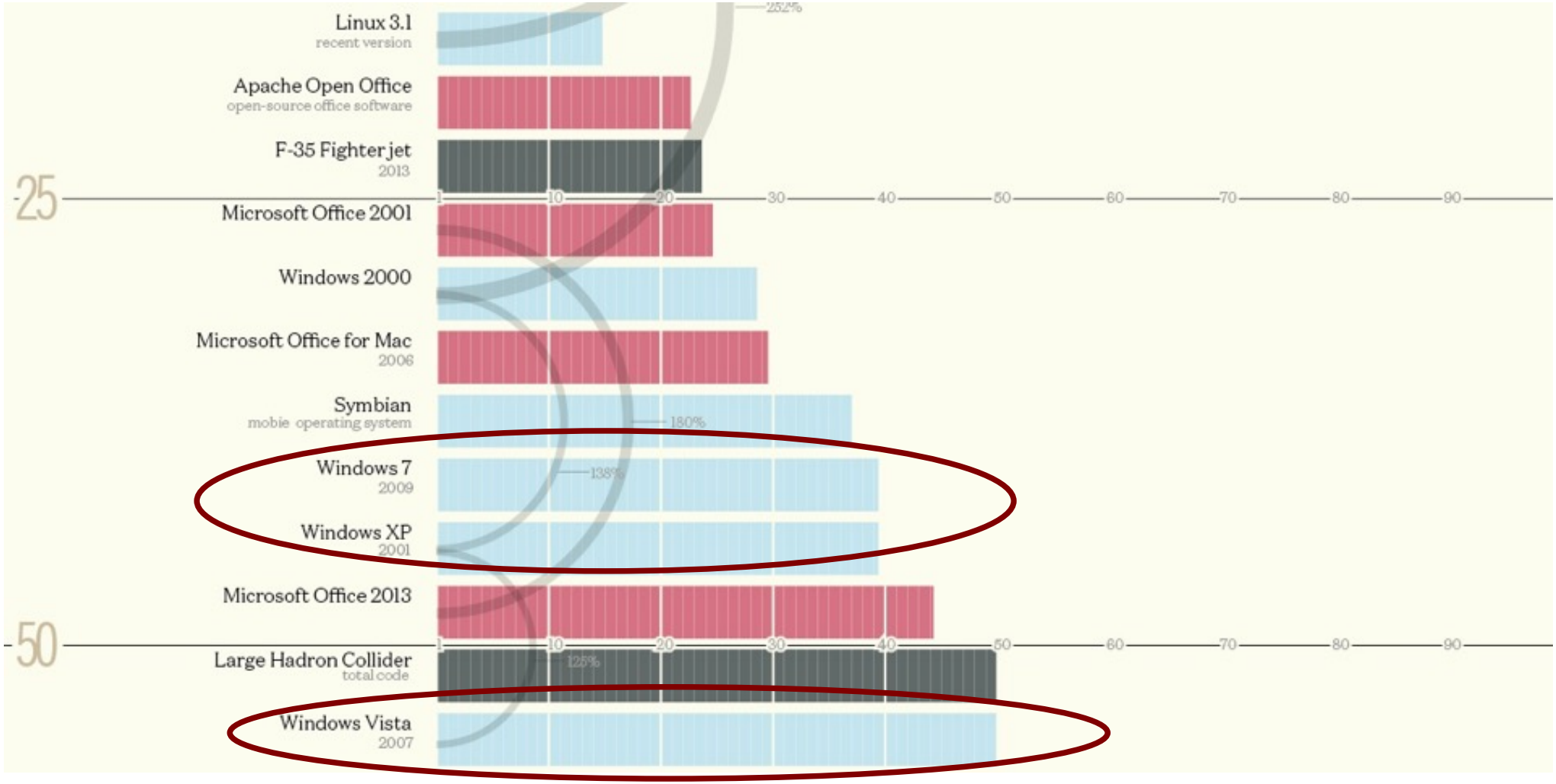


phonearena.com/news/Lines-of-code-How-our-favorite-apps-stack-up-against-the-rest-of-tech_id49281

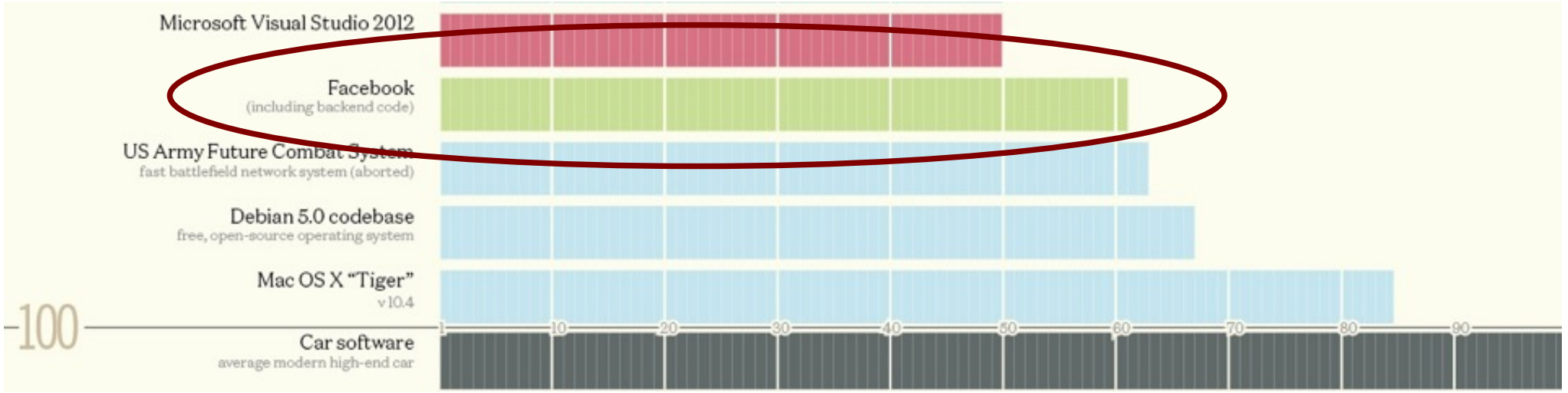
WHY MODELING SOFTWARE?



WHY MODELING SOFTWARE?

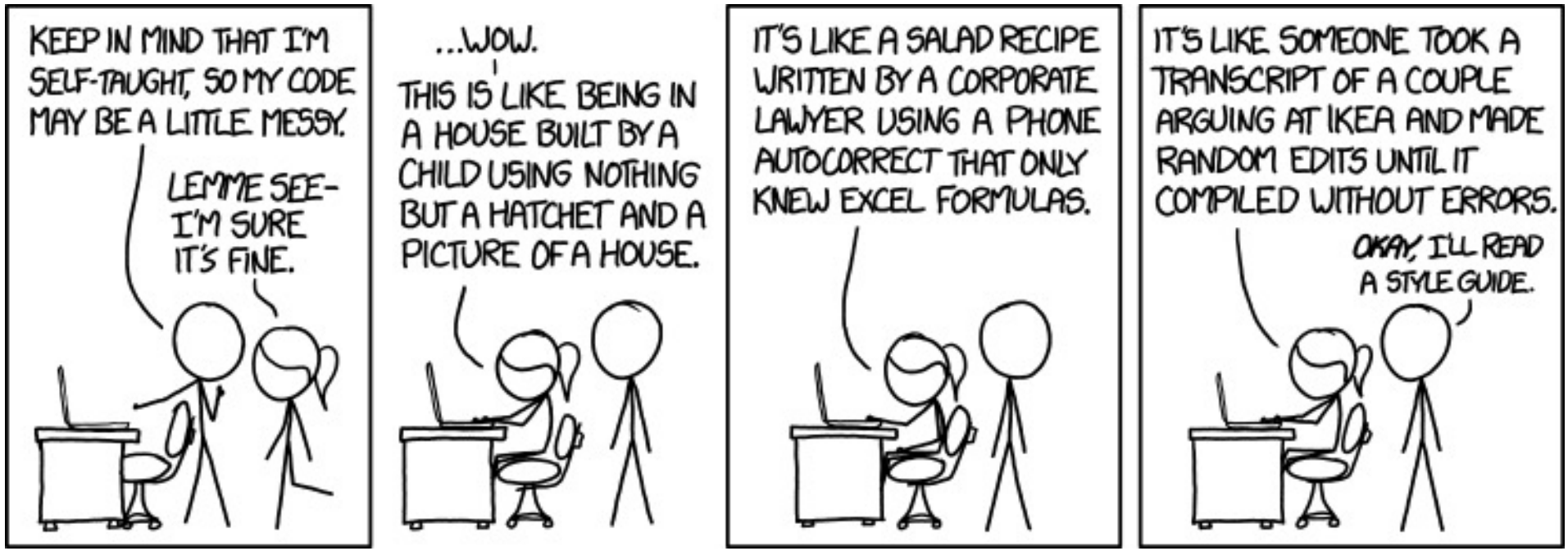


WHY MODELING SOFTWARE?



WHY MODELING SOFTWARE?

- **Complexity** is increasing
- One programmer **cannot manage** huge amounts of code
- **No easy understandable** to other developers



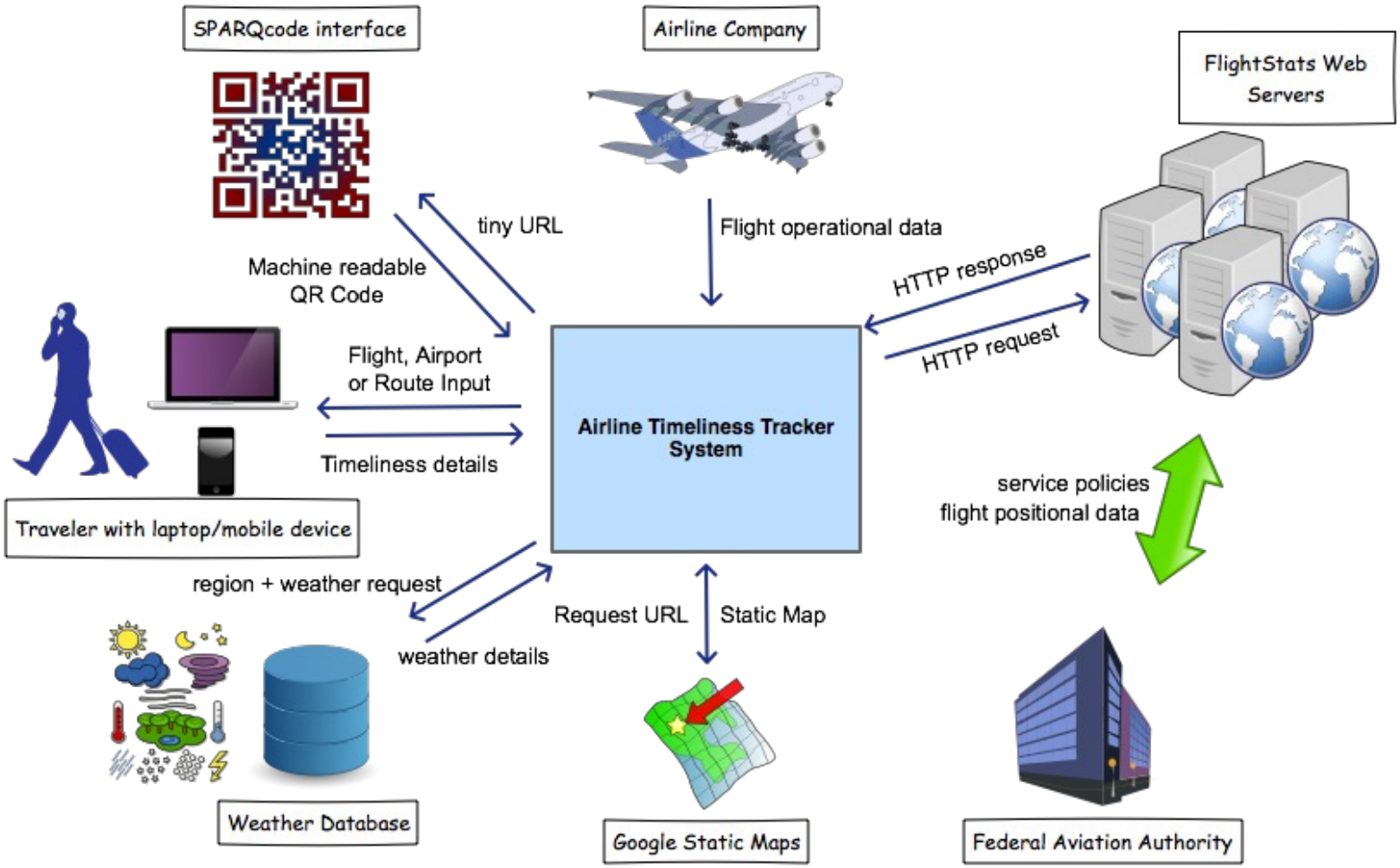
http://varianceexplained.org/programming/bad-code/

Modeling deals with complexity

MODELING A SYSTEM

1. Context Modeling -> System Context Diagram

How the system interacts with the environment e.g., actors, inputs, outputs



urabhpendse.com/csc510/timelinesstracker/site/context.html

1. CONTEXT MODELING

Actors:

Entity outside the software system

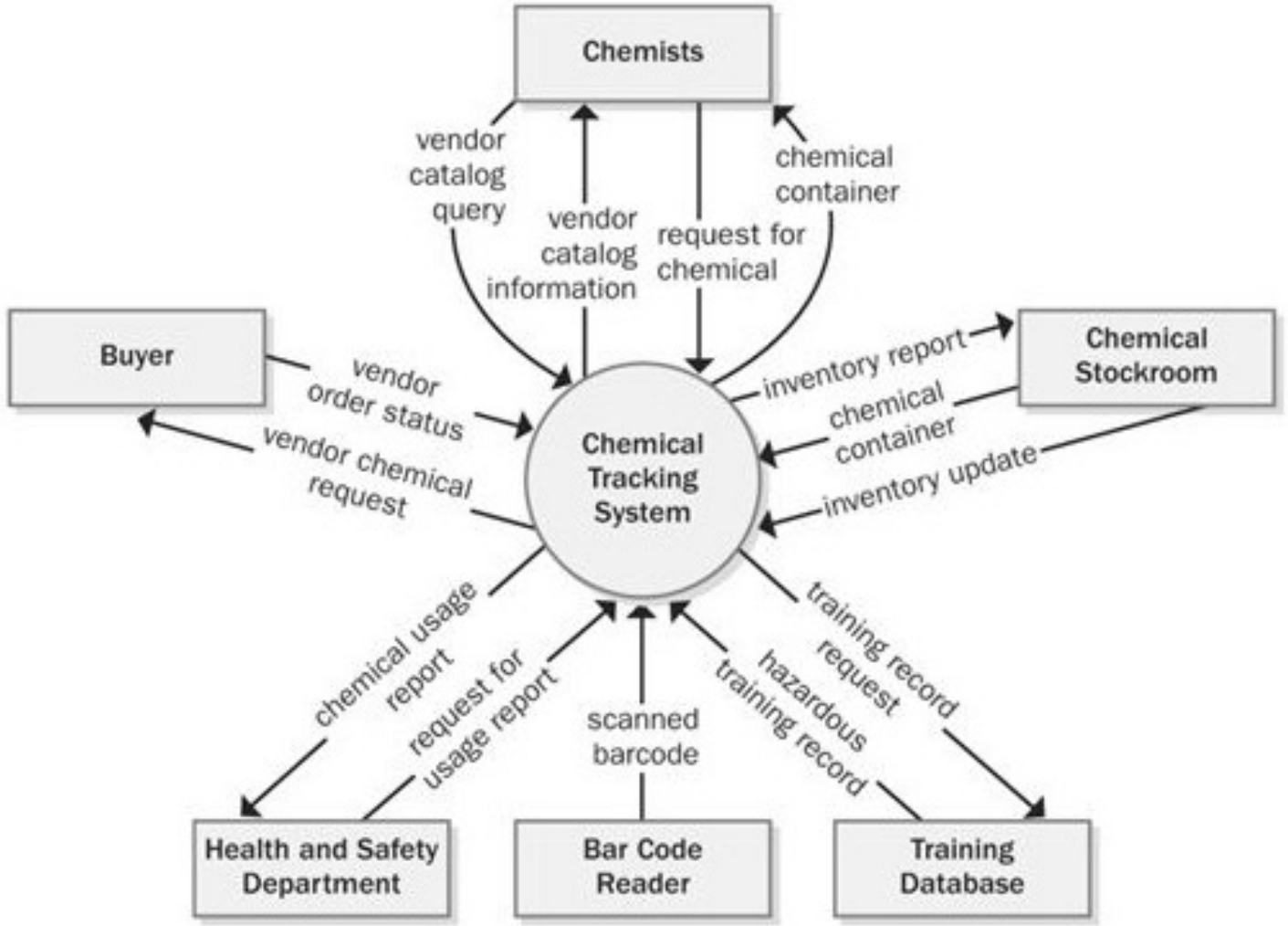
- Interacts with the system
- Another software, human, a hardware device, etc.

Represents a coherent **role** played by users

- Teacher, student, database -> library system
- A user might take on more than one role
- An actor might represent more than one user

1. CONTEXT MODELING

Highest level view of a system

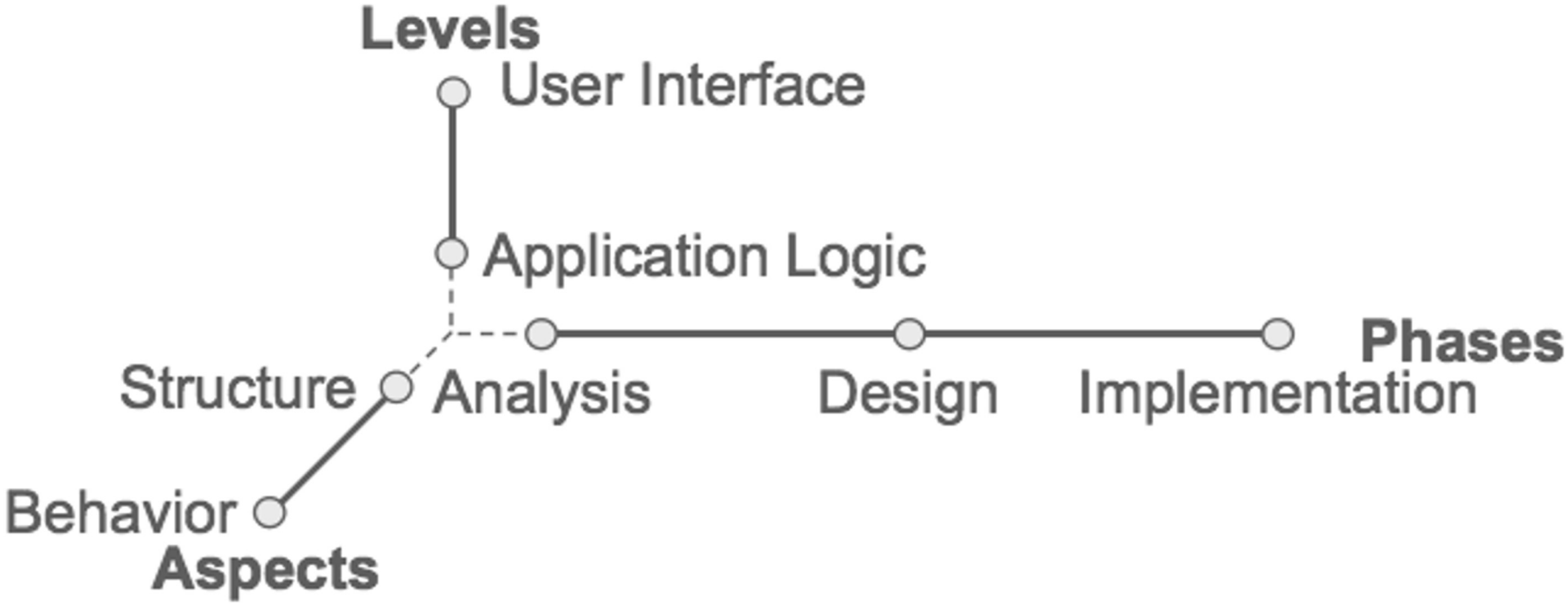


ndreaangella.com/2013/01/software-requirements-part-3/

MODELING A SYSTEM

Requirements for software application modeling

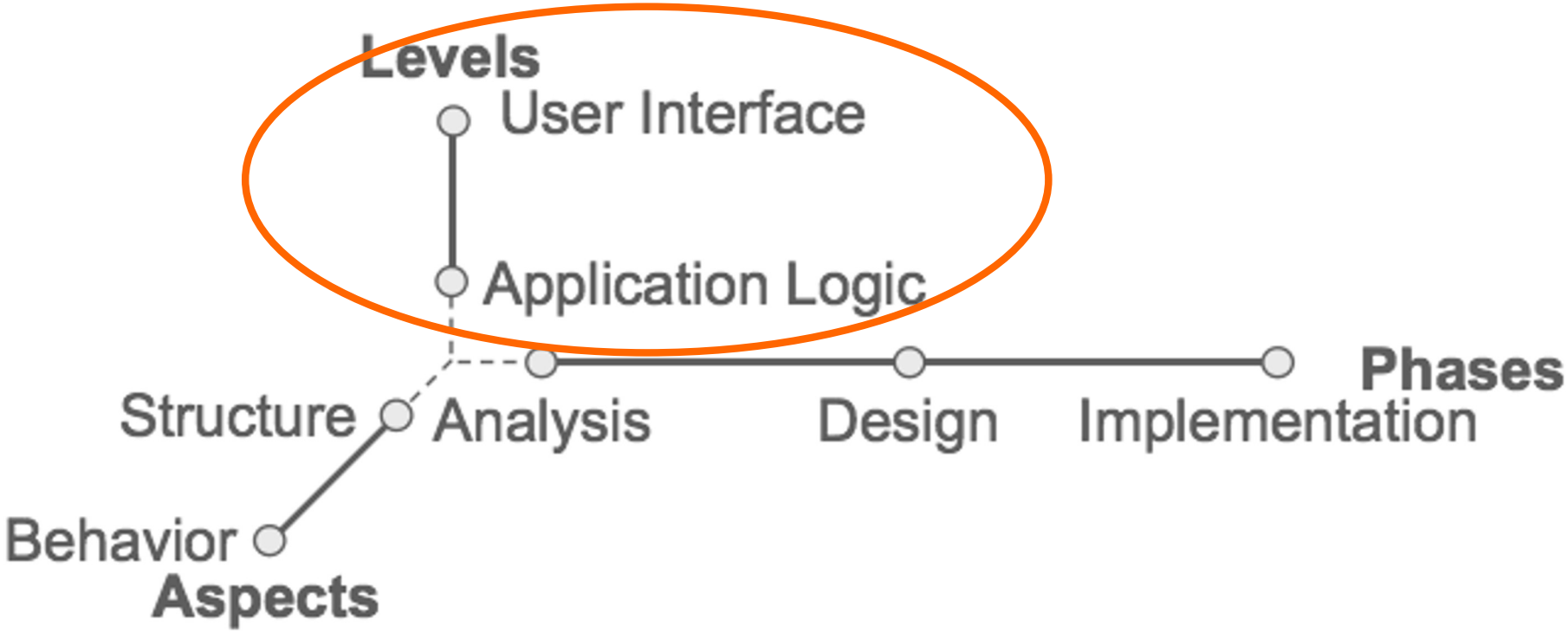
[Kappel et al. 2006, p. 40]



MODELING DIMENSIONS

Requirements for software application modeling

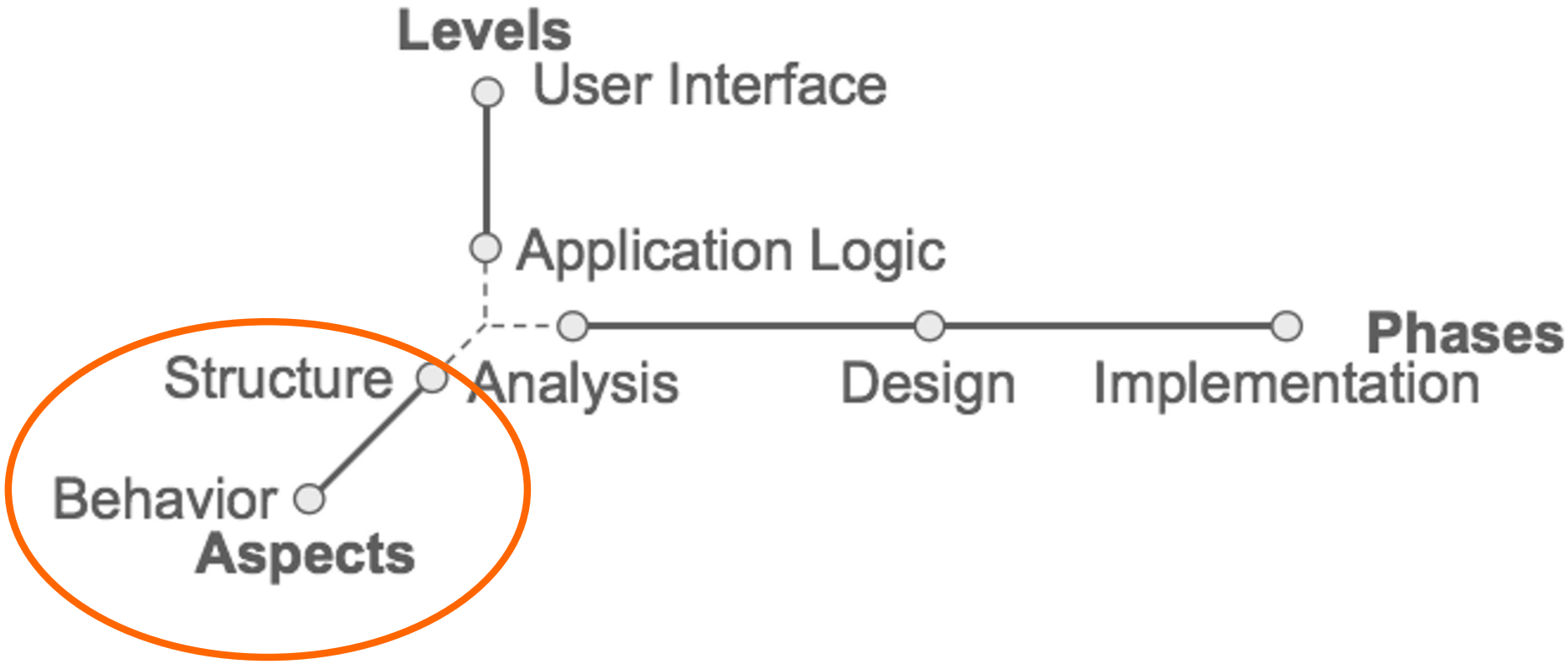
[Kappel et al. 2006, p. 40]



MODELING DIMENSIONS

Requirements for software application modeling

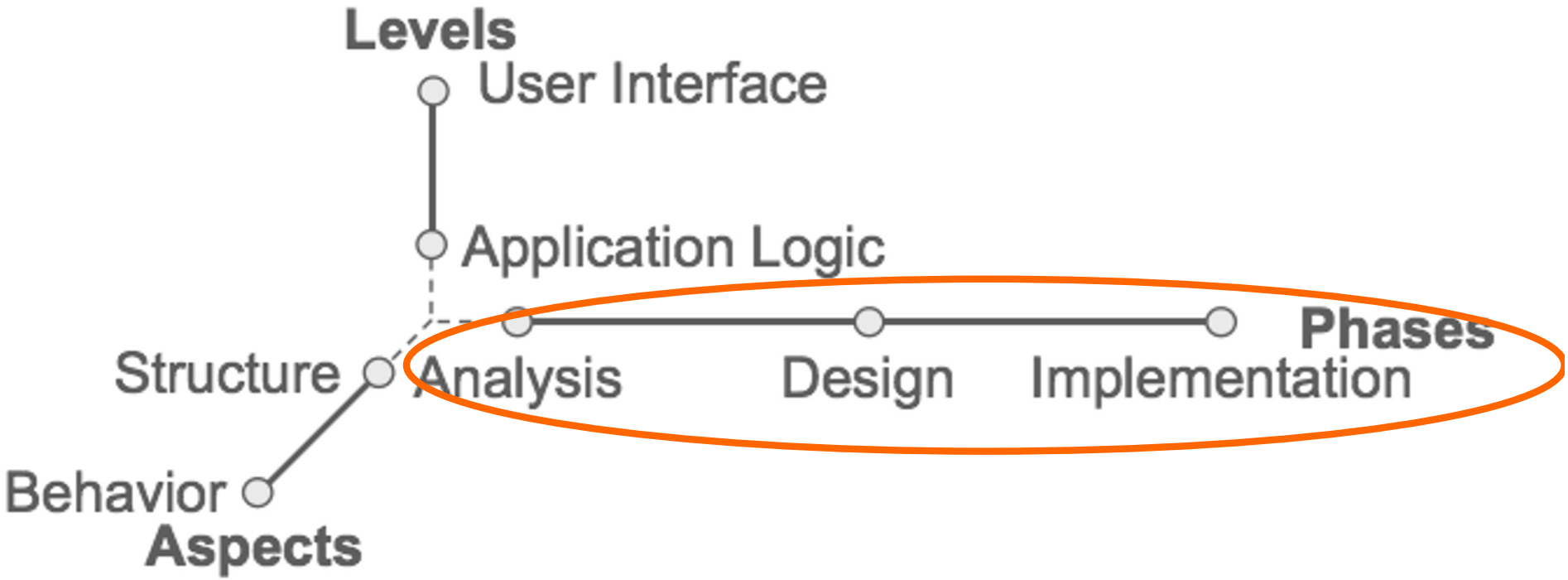
[Kappel et al. 2006, p. 40]



MODELING DIMENSIONS

Requirements for software application modeling

[Kappel et al. 2006, p. 40]



UNIFIED MODELING LANGUAGE (UML)

- It is a standard language for writing software blueprints [Booch99]
- Used to visualize, specific, construct and document the artifacts of a software-intensive system [Booch99]
- Tool to support systematic design
  - It is possible to model 80% of most systems by using 20% UML

TYPES OF DIAGRAMS

- **Functional modeling ->** Functionalities of the system
  - Use case diagrams

- **Static/Structural modeling ->** Static aspects of the system that do not change during the program execution
  - Class and object diagrams

- **Dynamic/Behavioral modeling ->** Dynamic aspects of the system that happens during program execution
  - Interaction, activity and state machine diagrams

SUMMARY

High-level capture of requirements

- **Use Case Diagram**

Identify major objects and relationships

- **Class diagram (object diagram)**

Create scenarios of usage

- Interaction Diagrams -> m. Sequence Diagram

Generalize scenarios to describe behavior

- **Sate Diagram**
- **Activity Diagram**

User Interface -> Testing some functionalities

- **Sketching and Prototyping**