Design [Software]

Ack: Some slides taken from Andre ☺

Tip:

[Pre-Condition: Thinking]
Design
[Post-Condition: Doing]

What is Design?


AWD

case study

Design Case Study
Handheld Interactive Device
For Collaborative Learning among Kids
by
Kumar Anchal
Industrial Design Centre (IDC), IIT Bombay

Design Case Study
The India Post Box Re-design
Products for the Public Domain
by
Prof. B. K. Chakravarthy
Industrial Design Centre (IDC), IIT Bombay

Design Case Study
Design of an Electric Cart
Vehicle Design for Malls
by
Shweta Suthar
Industrial Design Centre (IDC), IIT Bombay

Design Case Study
Akshaya Patra
Inexhaustible vessel
by
Prof. Ravi Mokashi Punekar and Mr. K. K. Balakrishnan
Department of Design, IIT Guwahati

showcase

Design Showcase
Lemon Design
Strategic Branding and Design
by
Professor Ravi Mokashi Punekar
Department of Design, IIT Guwahati

Design Showcase
Anuj Prasad - Desmania
Product Design Firm
by
Professor Ravi Mokashi Punekar
Department of Design, IIT Guwahati

Design Showcase
Elephant Design
Strategic Design and Brand Consultancy Firm
by
Professor Ravi Mokashi Punekar
Department of Design, IIT Guwahati

Design Showcase
Antony Lopez
Independent Design Studio
by
Antony Lopez

gallery

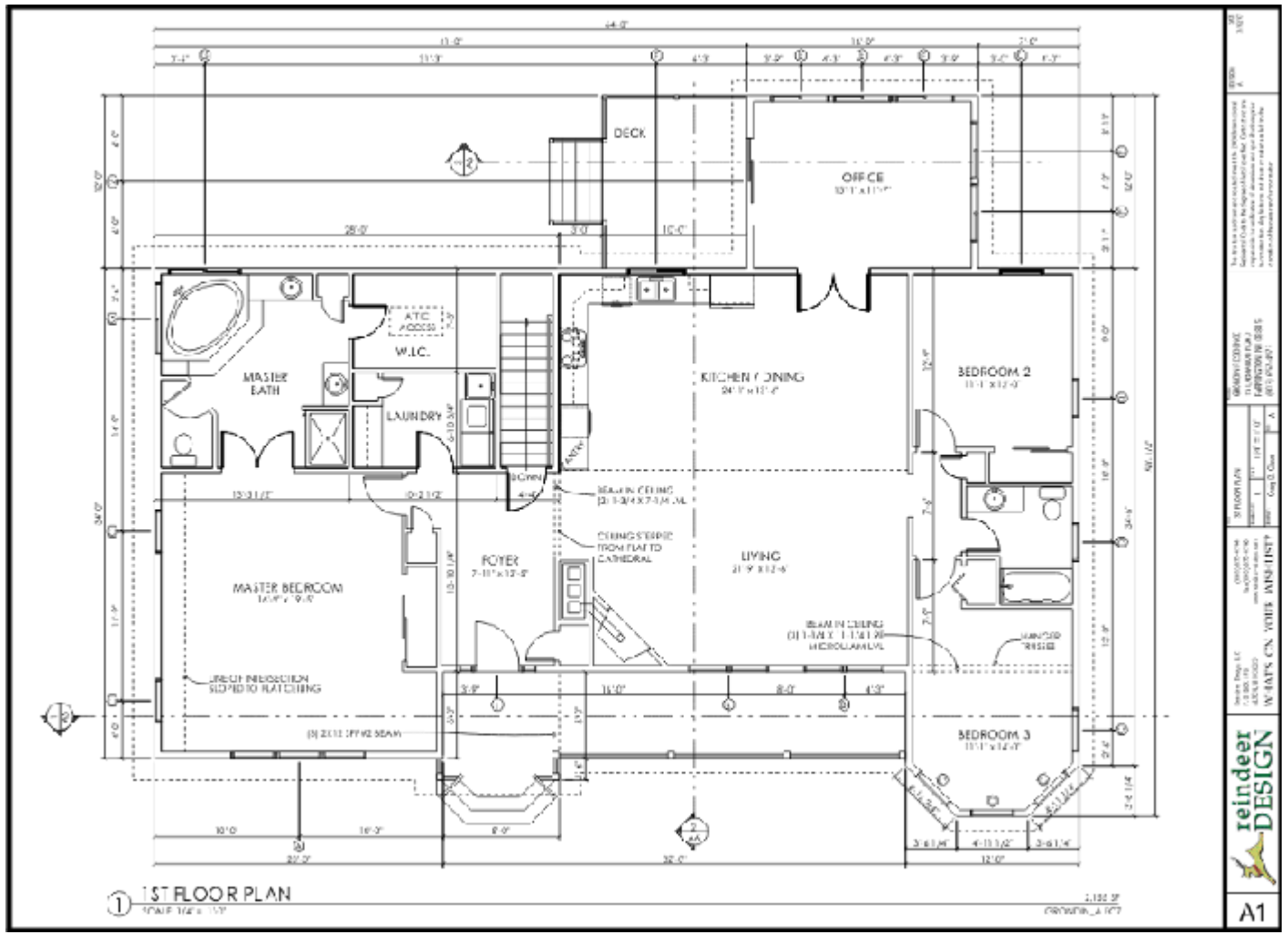
Gallery
Viva Carnaval Goa
Popular Portuguese Carnival
by
Sunil Mahajan
Industrial Design Centre (IDC), IIT Bombay

Gallery
Classic Logos of India
Classic Logos of India
by
Nanki Nath, Ph. D student and
Prof. Ravi Poovaiah
Industrial Design Centre, IIT Bombay

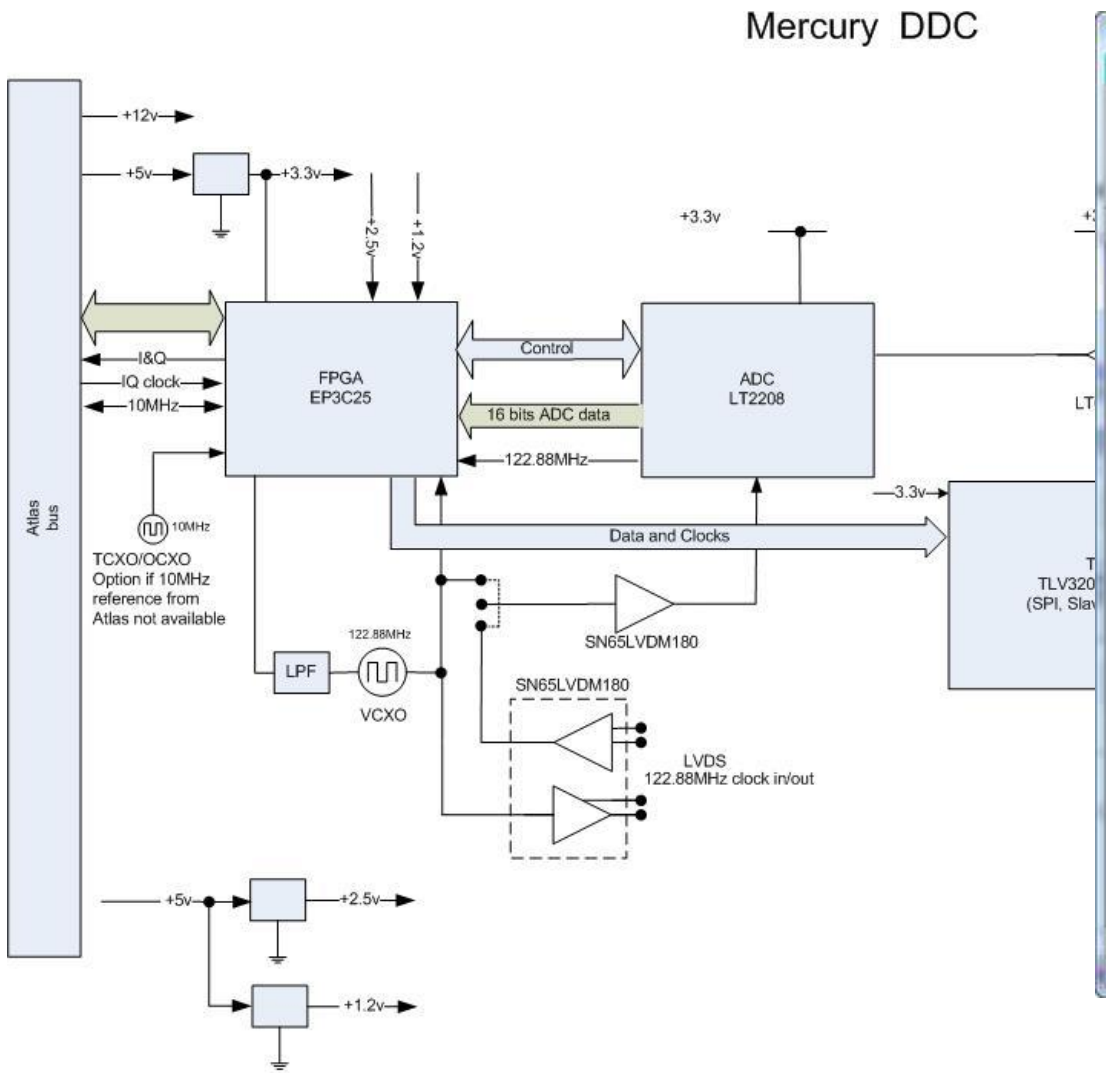
Gallery

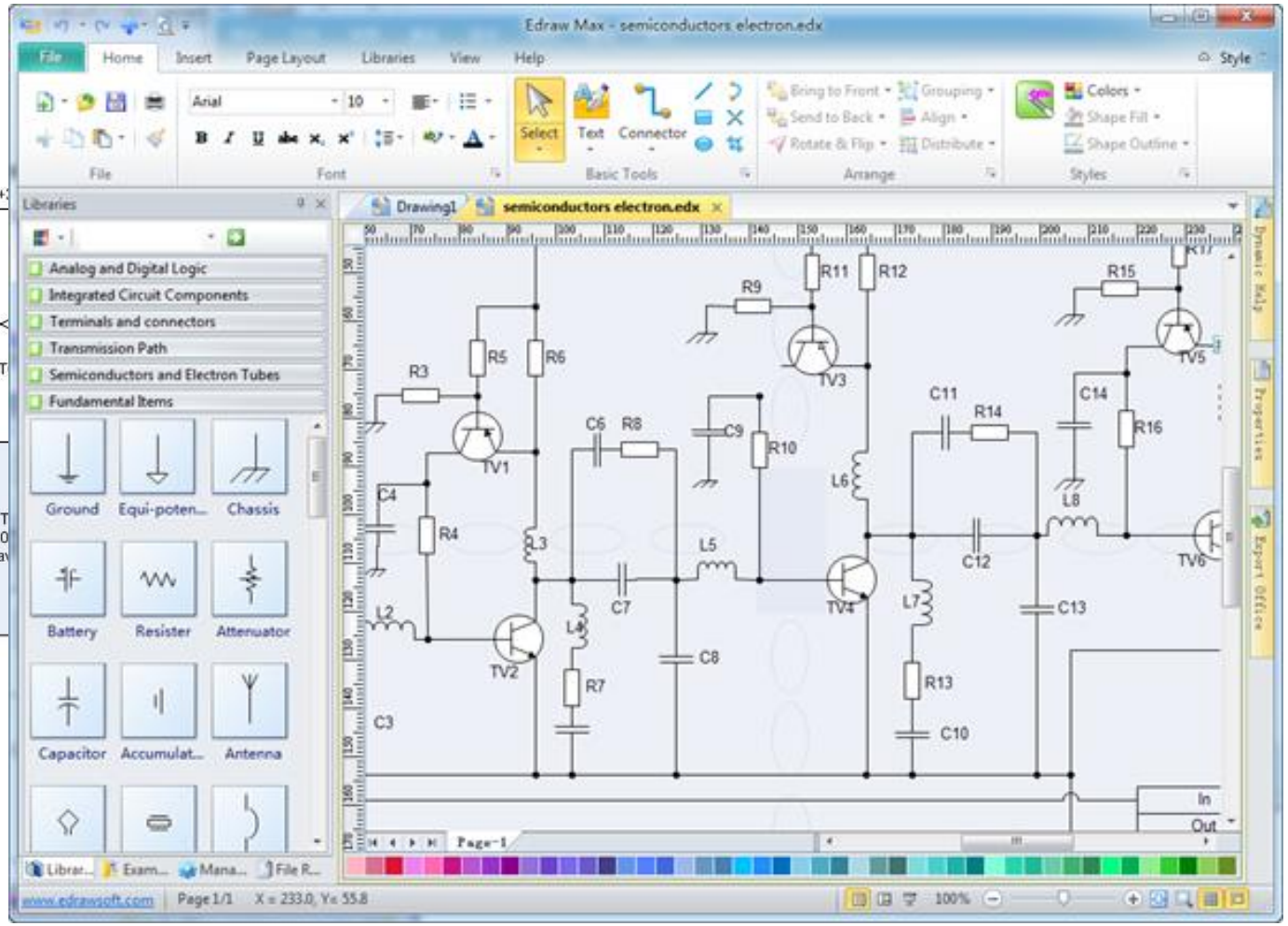
Gallery

Floor plan

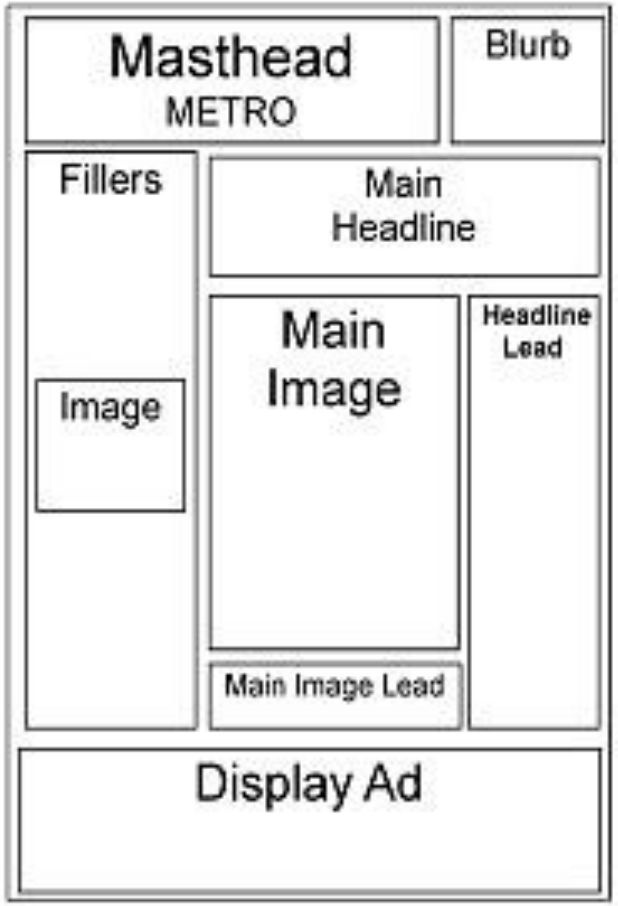

DECK
OFFICE
ATTIC ACCESS
W.I.C.
MASTER BATH
LAUNDRY
KITCHEN / DINING
BEDROOM 2
FOYER
LIVING
MASTER BEDROOM
BEDROOM 3
1 1ST FLOOR PLAN
reindeer DESIGN
A1

Electric Circuits





Page layout – Design?



Mechanical engineering diagram

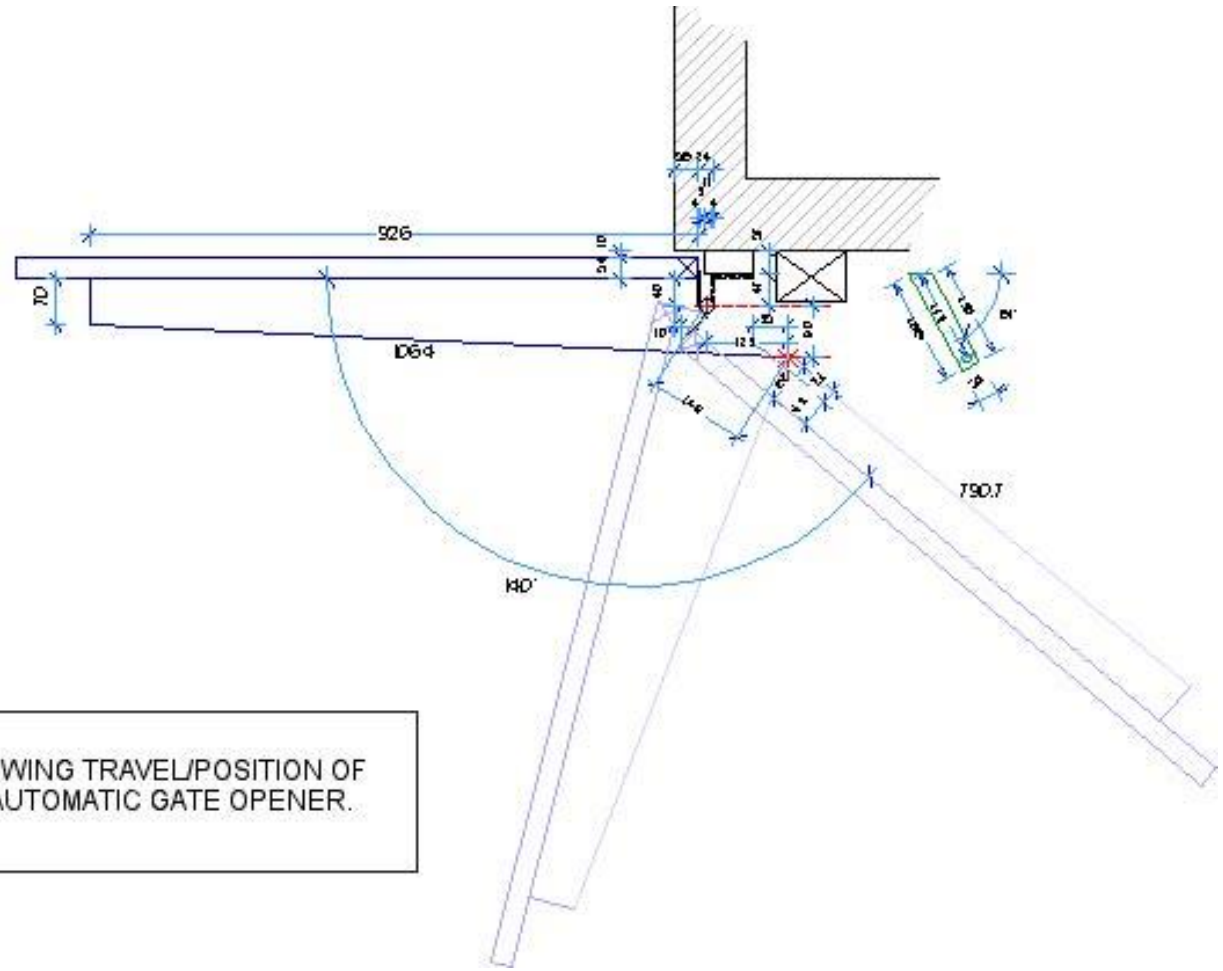
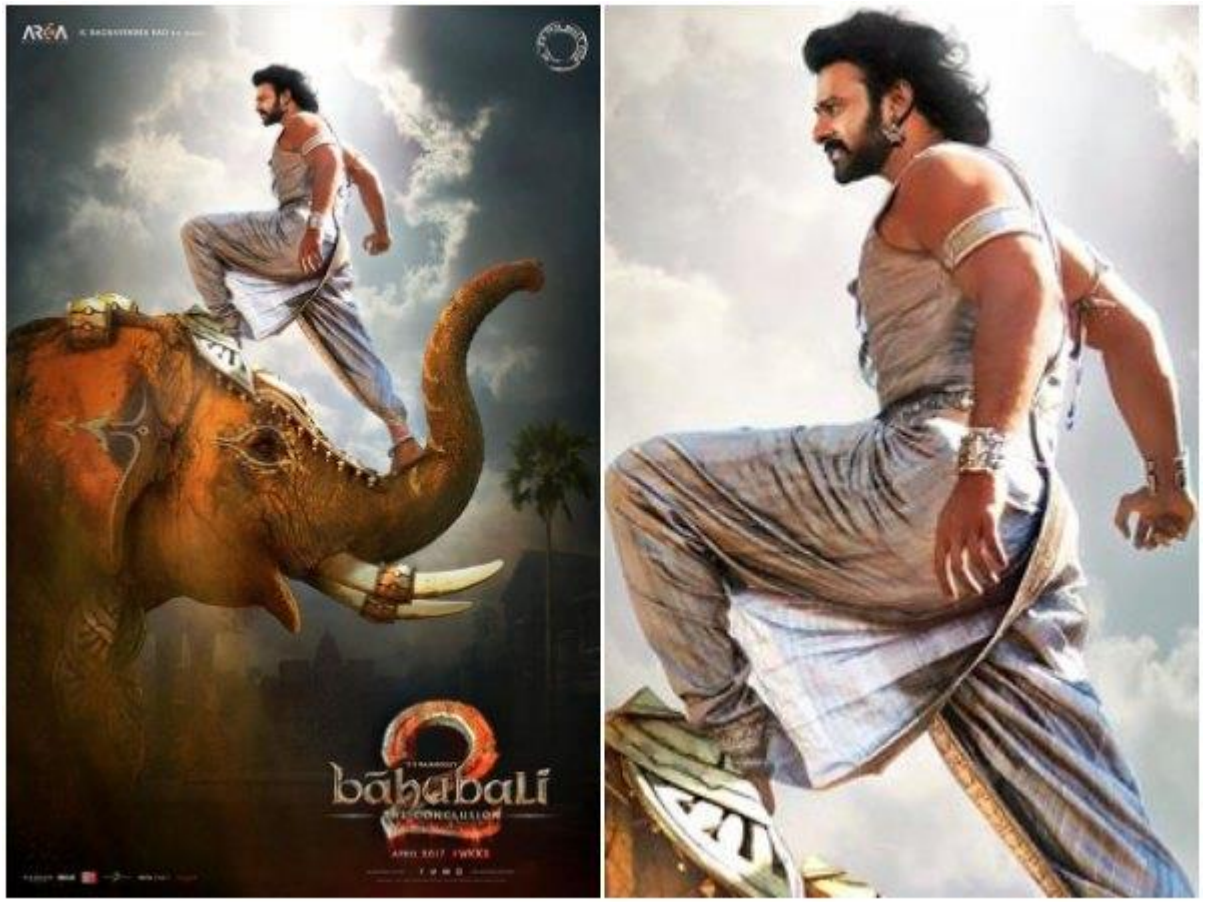
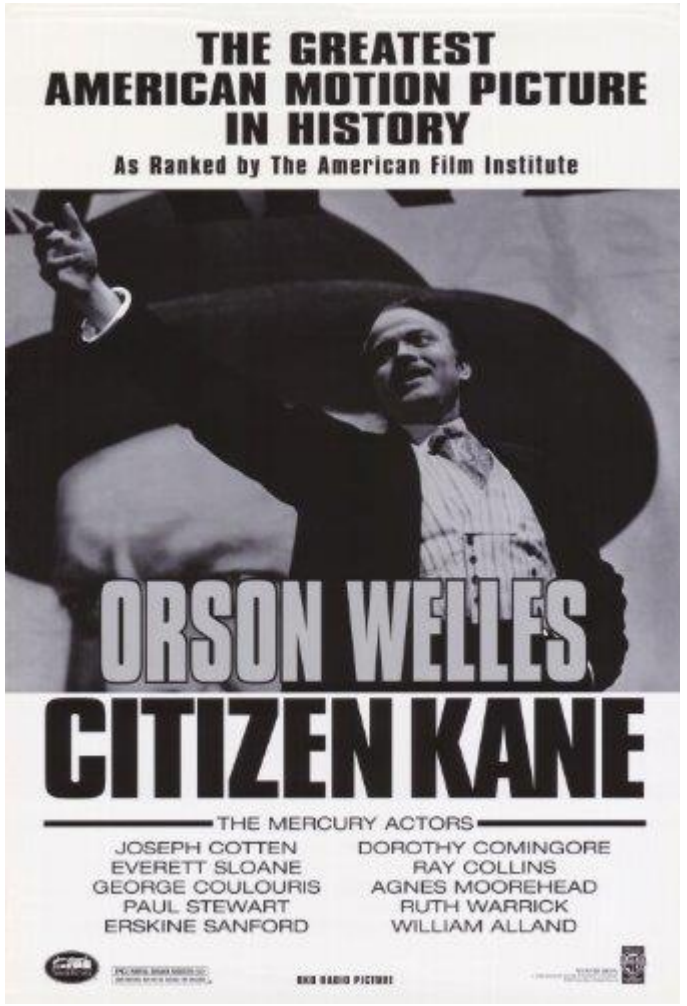


DIAGRAM SHOWING TRAVEL/POSITION OF SWING TYPE AUTOMATIC GATE OPENER.

Movies/Series?

Movies?





Credits to Original Creators

Games?


DOOM
FARCRY
Peggle
BORDERLANDS
Little BigPlanet
POKÉMON
grand theft auto
SUPER MARIO GALAXY
STREET FIGHTER
PSYCHONAUTS
SCRIBBLENAUTS
BRÜTAL LEGEND

Activity 1

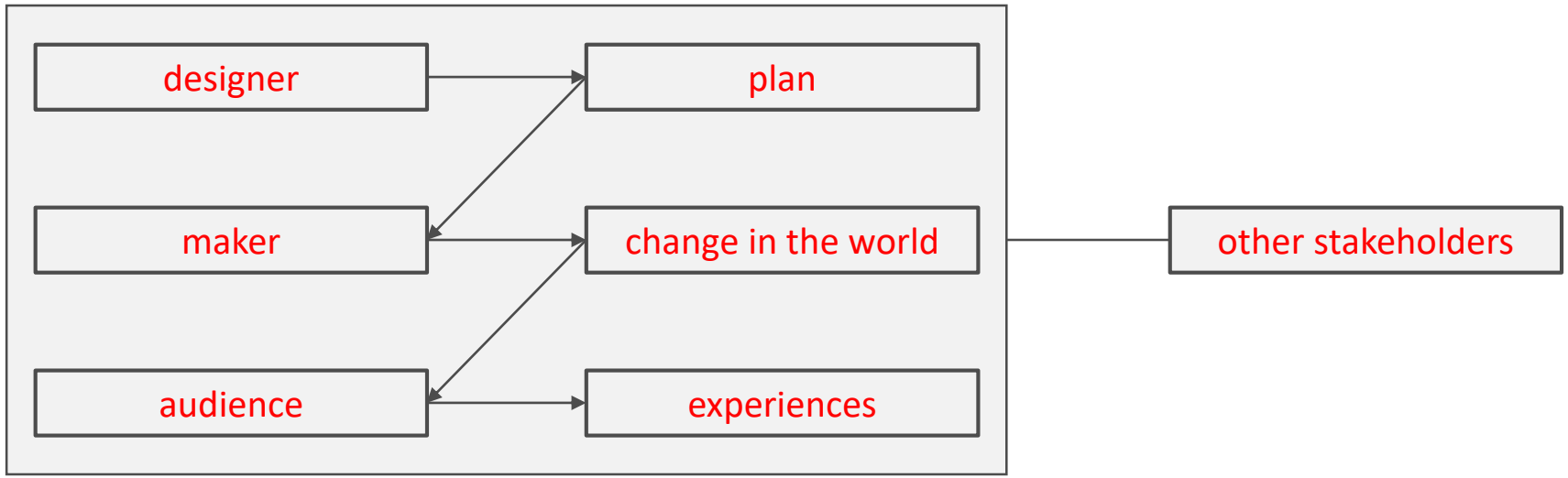
- Is Software Engineering like Movie Making? (Room 1)
  - Yes - ?
  - No - ?
- Is Software Engineering like Game Development? (Room 2)
  - Yes - ?
  - No - ?

- 12 minute breakout room – summarize your answers and type them in chat after done!

Movie Making & Design

So, what is *design* anyway?

Design

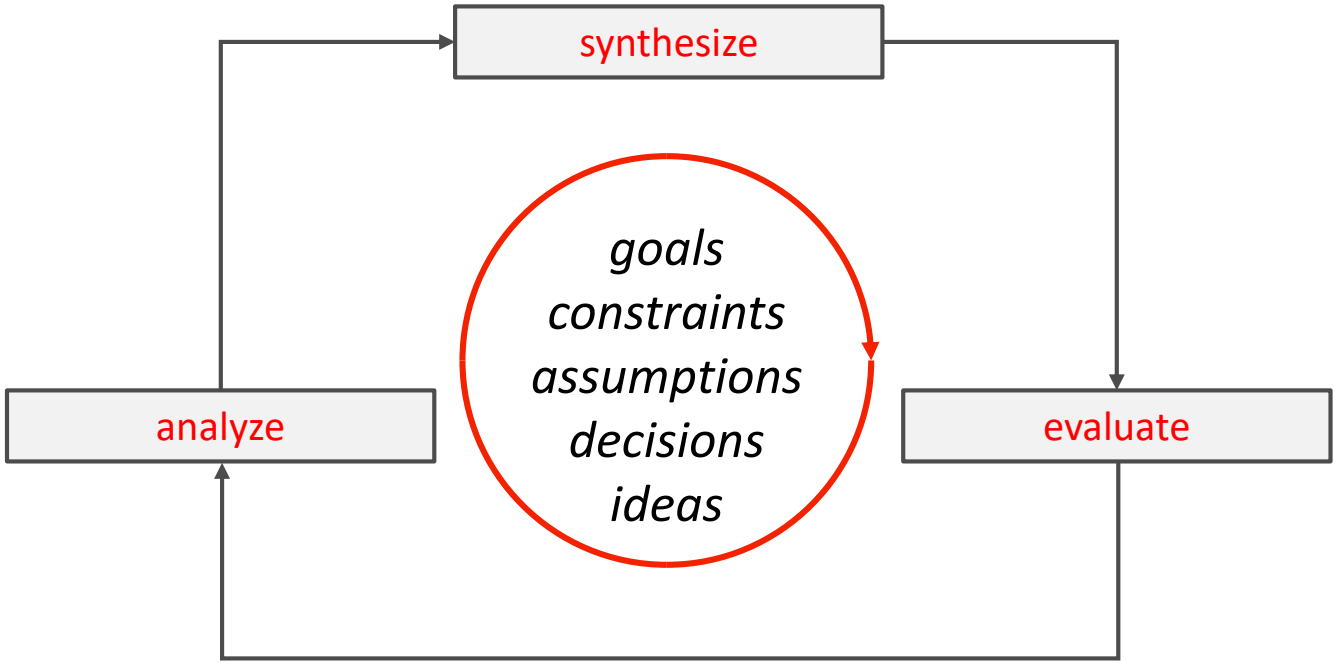


- To decide upon a plan for change in the world that, when realized, satisfies stakeholders – Andre

Defining design

- To initiate change in man-made things
- To plan or intend for a purpose
- To work out a solution in one’s mind
- The transition from possible solutions to a specific one
- *Creative unique decision making under uncertainity*
- …

Design cycle



Design work

- Design work represents the individual or collaborative activity of engaging with a design project at a detailed level
  - thinking
  - articulating context
  - analyzing alternative ideas
  - identifying constraints
  - making decisions
  - setting goals
  - …

Design is a wicked problem

- The problem is not understood until after the formulation of a solution
- Wicked problems have no stopping rule
- Solutions to wicked problems are not right or wrong
- Every wicked problem is essentially novel and unique
- Every solution to a wicked problem is a “one shot operation”
- Wicked problems have no given alternative solutions

To decide upon a plan for change in the world that, when realized, satisfies stakeholders

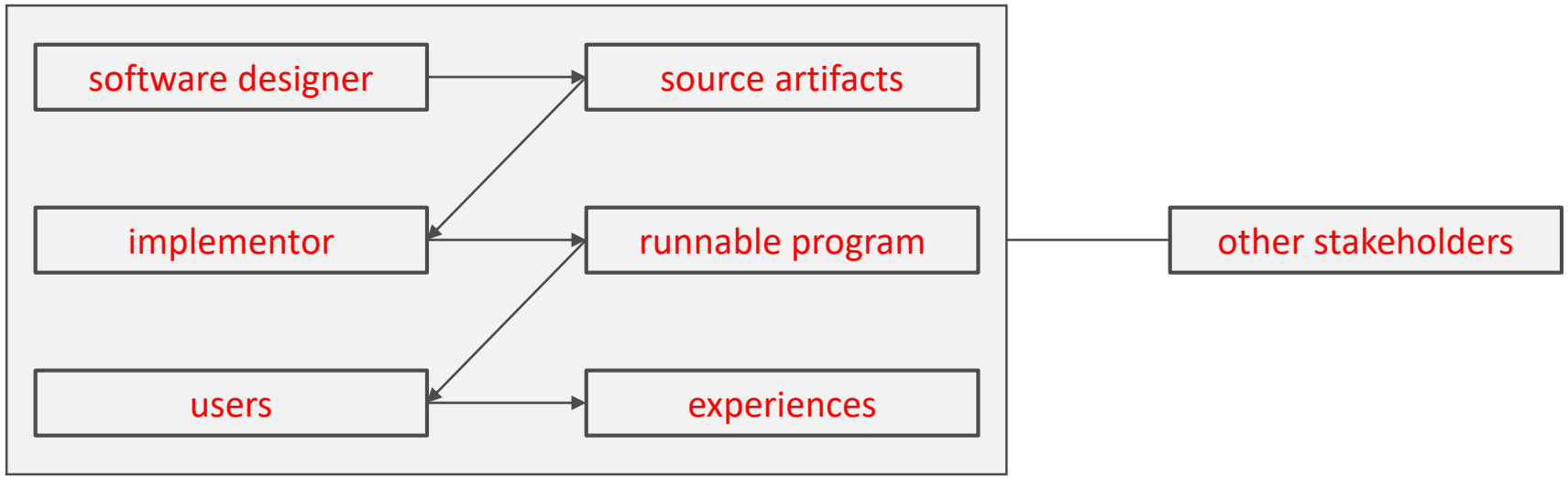
Design fields

- Architecture design
- Graphic design
- Fashion design
- Game design
- Chip design
- Car design
- Urban design
- Product design
- Interior design
- …

- Writing
- Painting
- Sculpting
- Music composition
- …

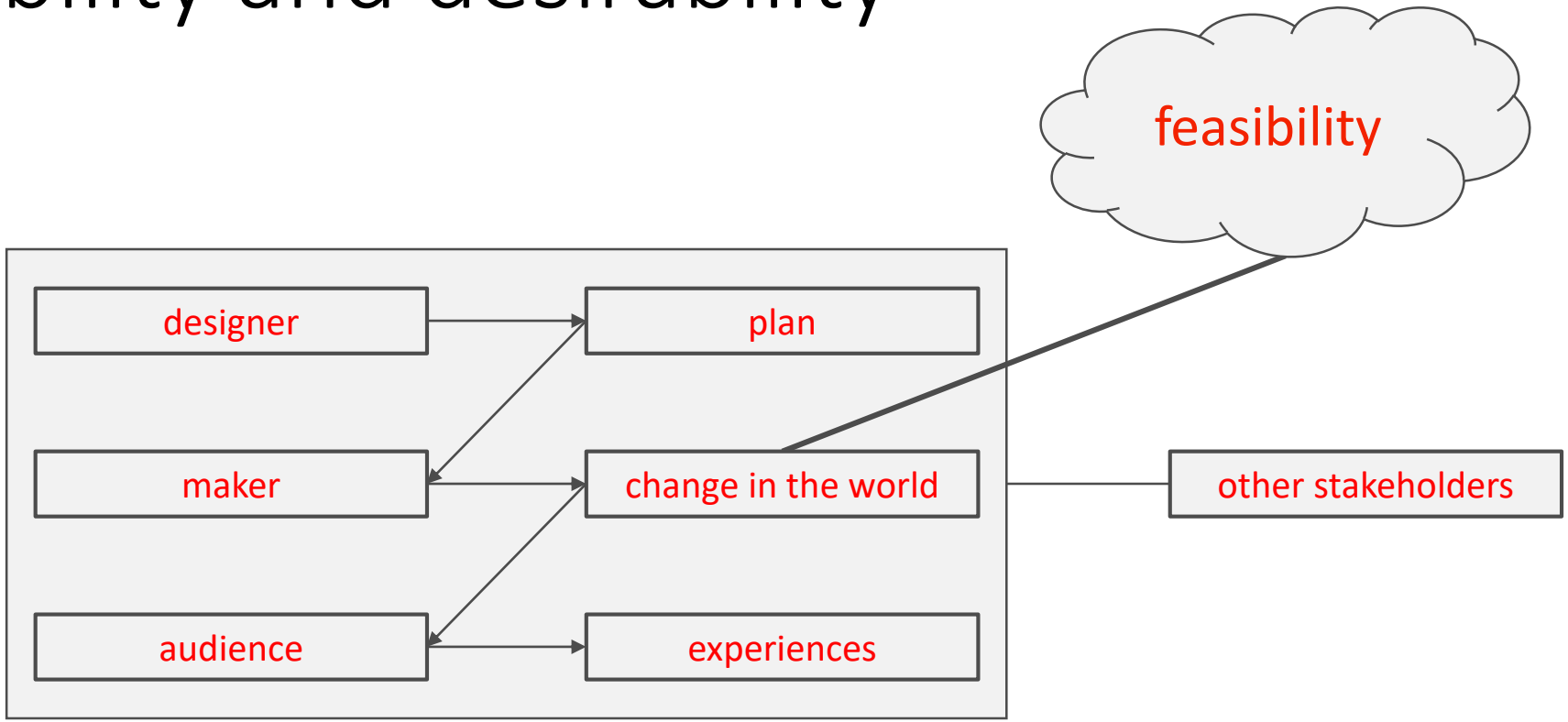
- Software design

Software design

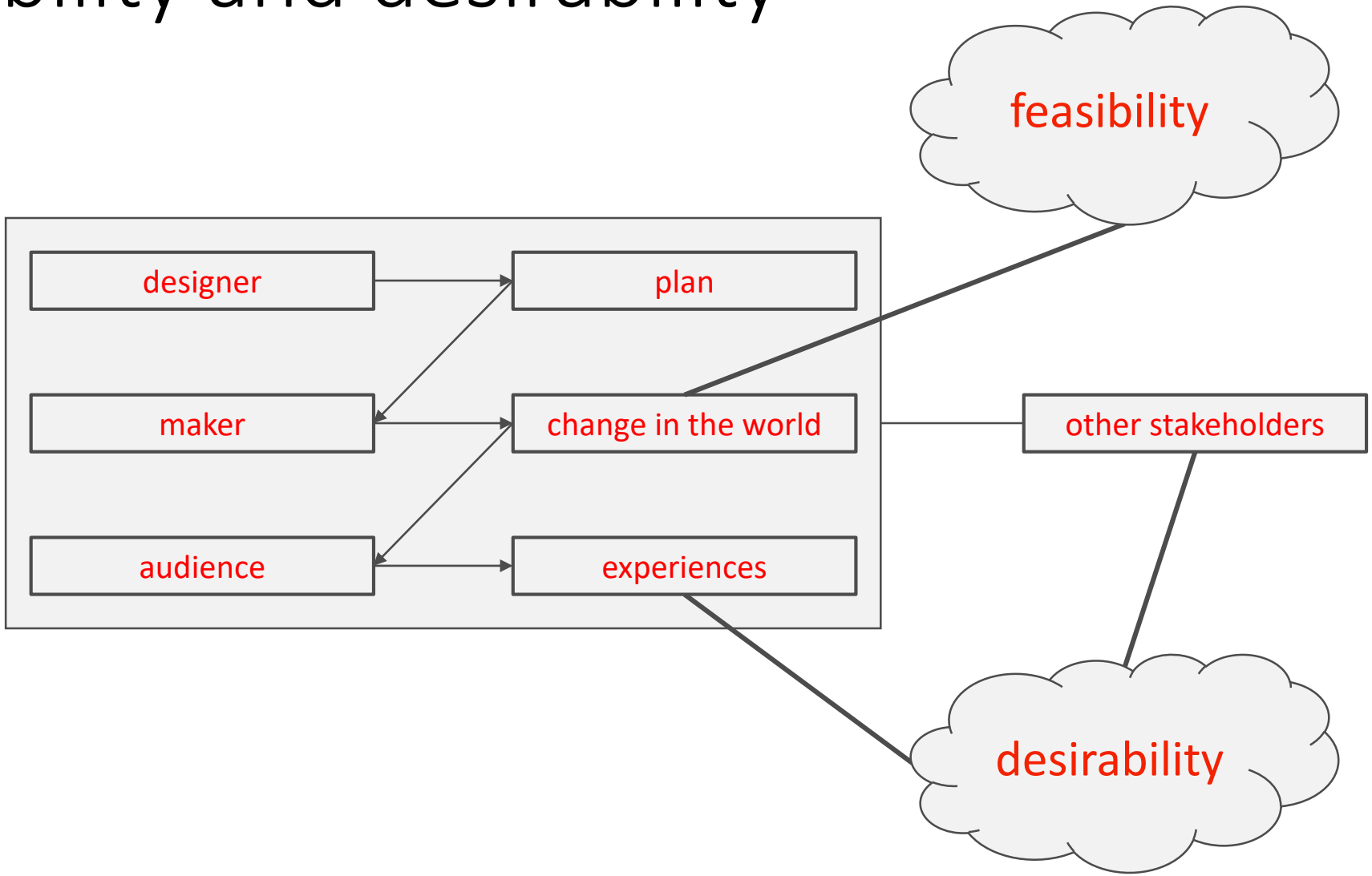


- A software design expresses a solution to a problem in programming language independent terms.

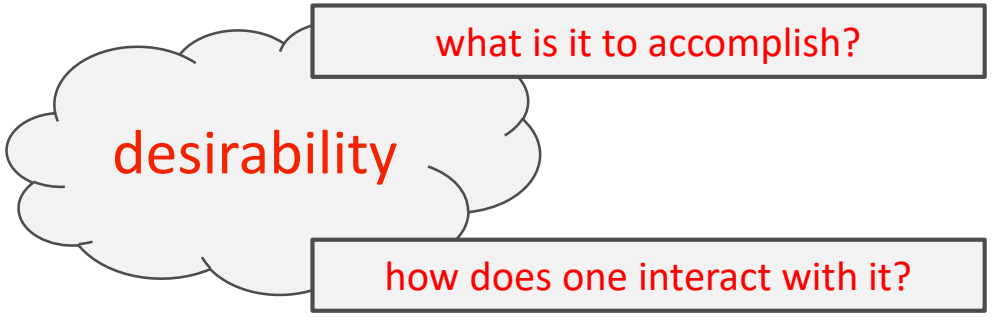
Feasibility and desirability

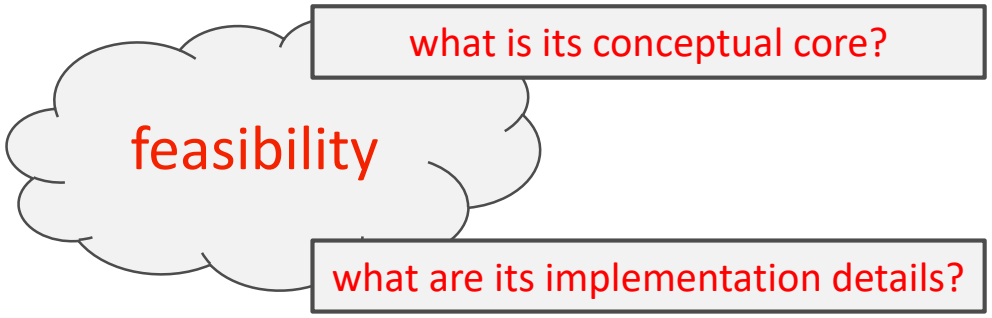


Feasibility and desirability

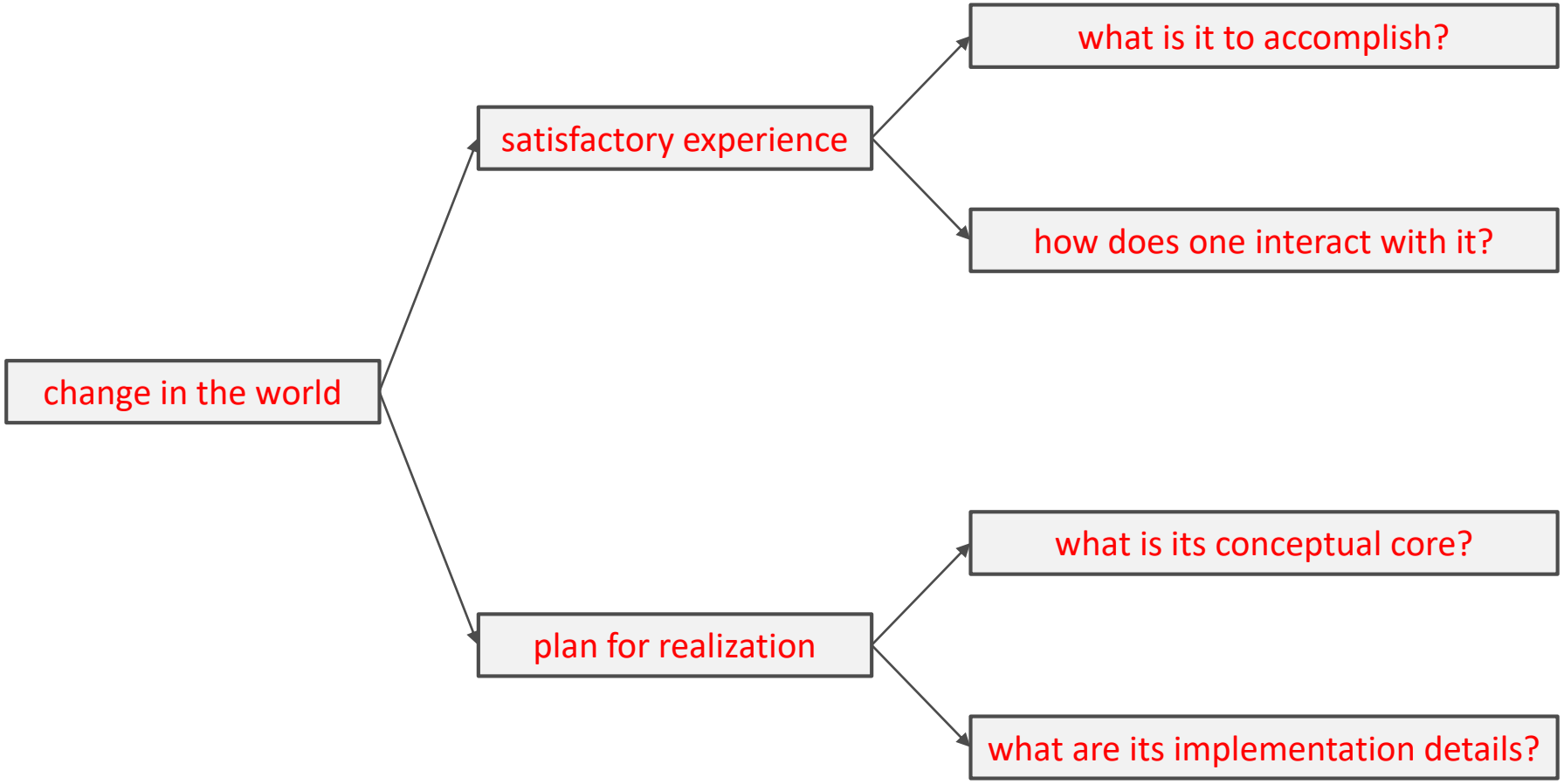


Four types of design

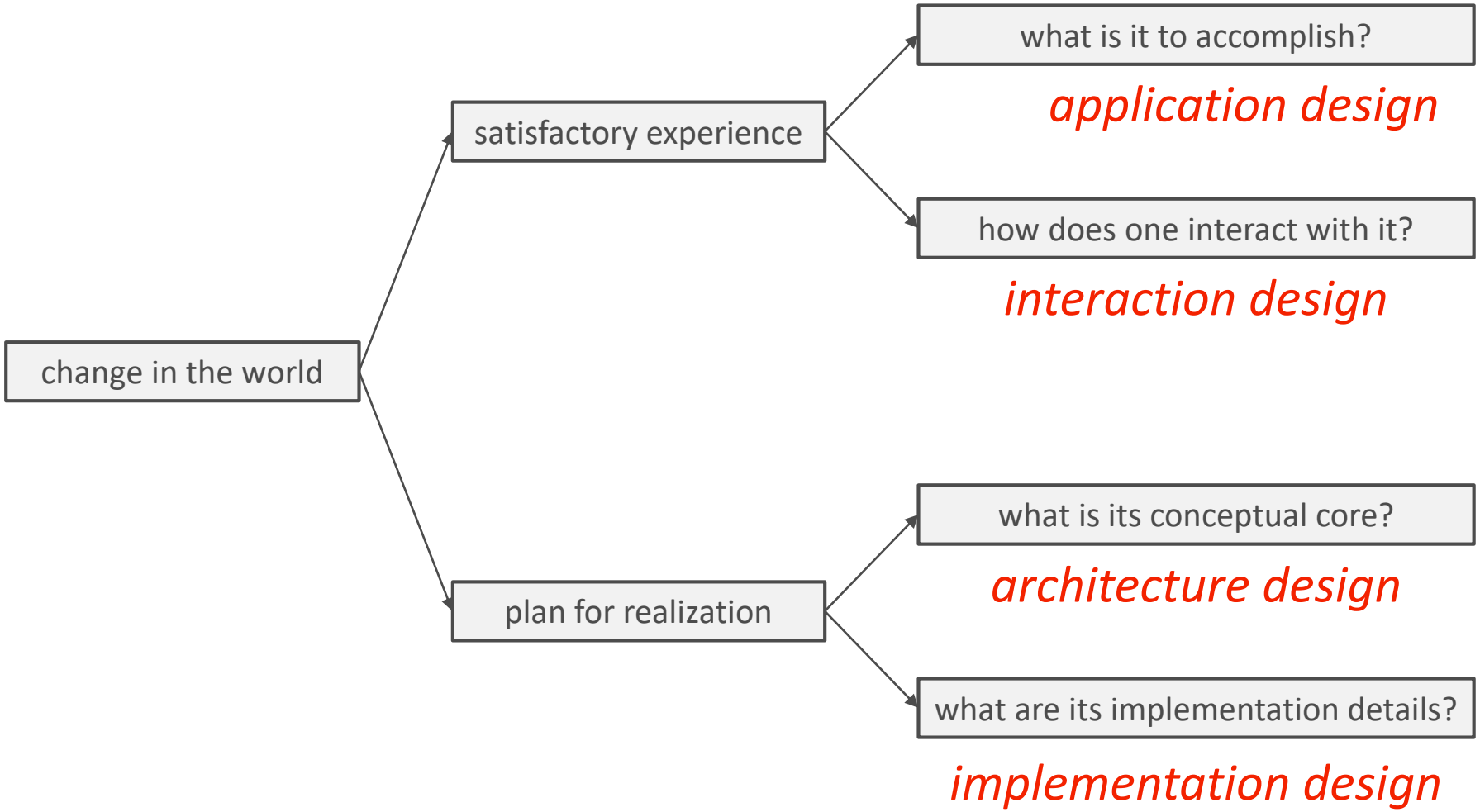




Four types of design

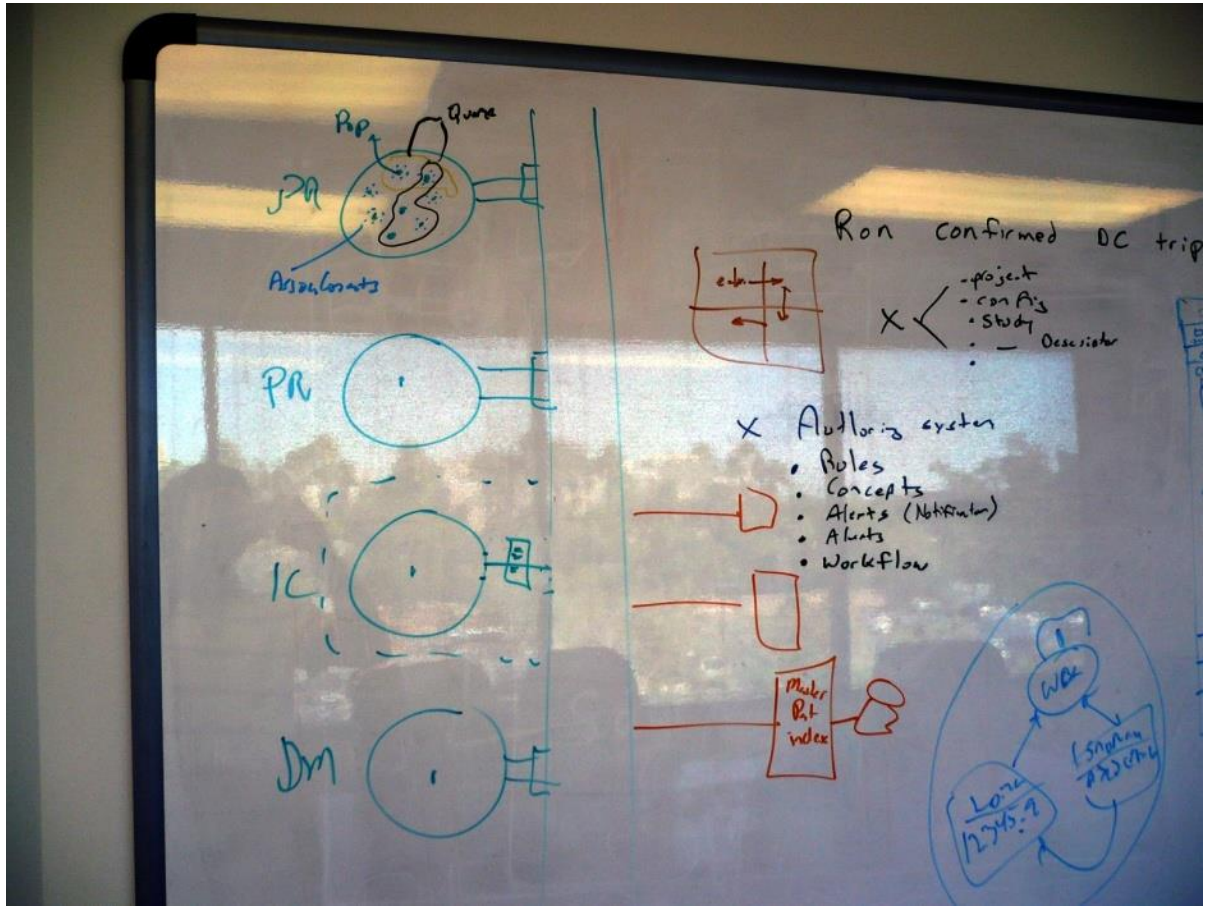


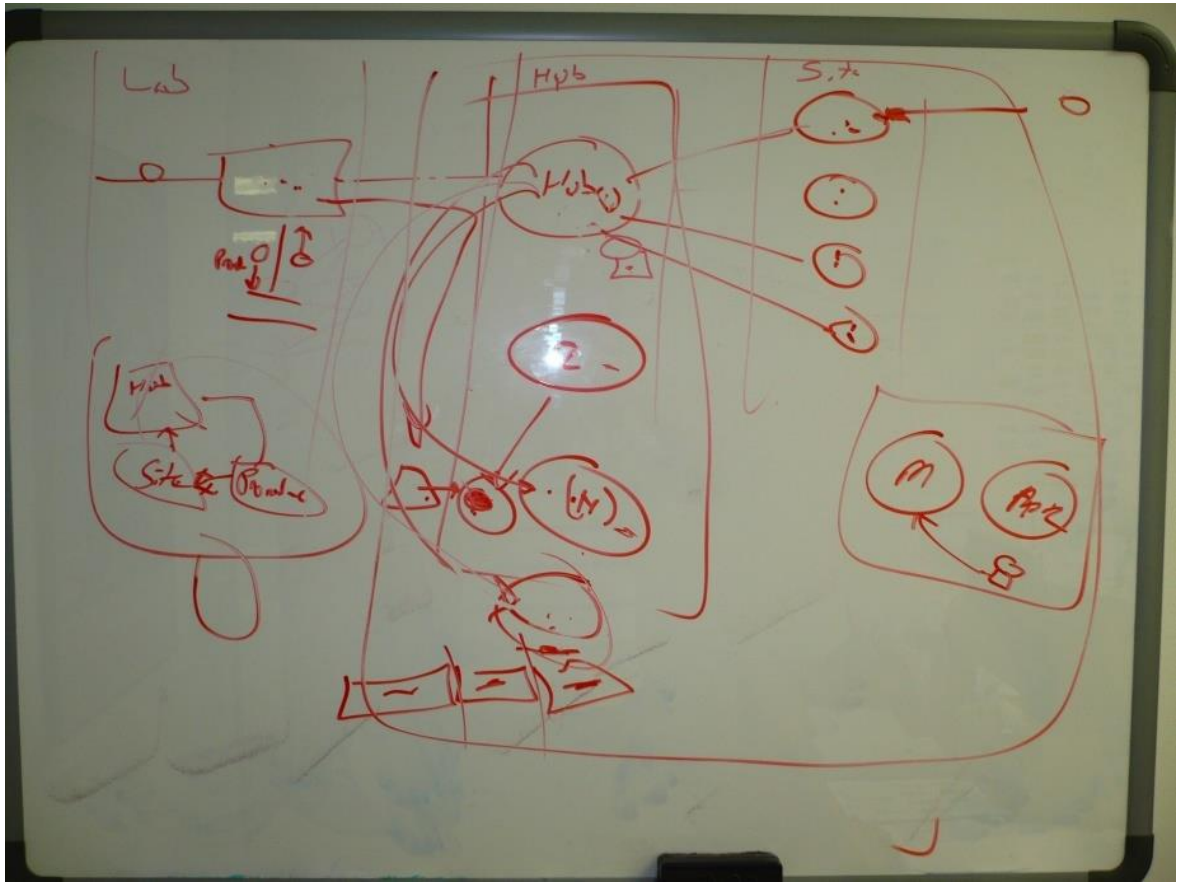
Four types of software design



Software Design - Examples

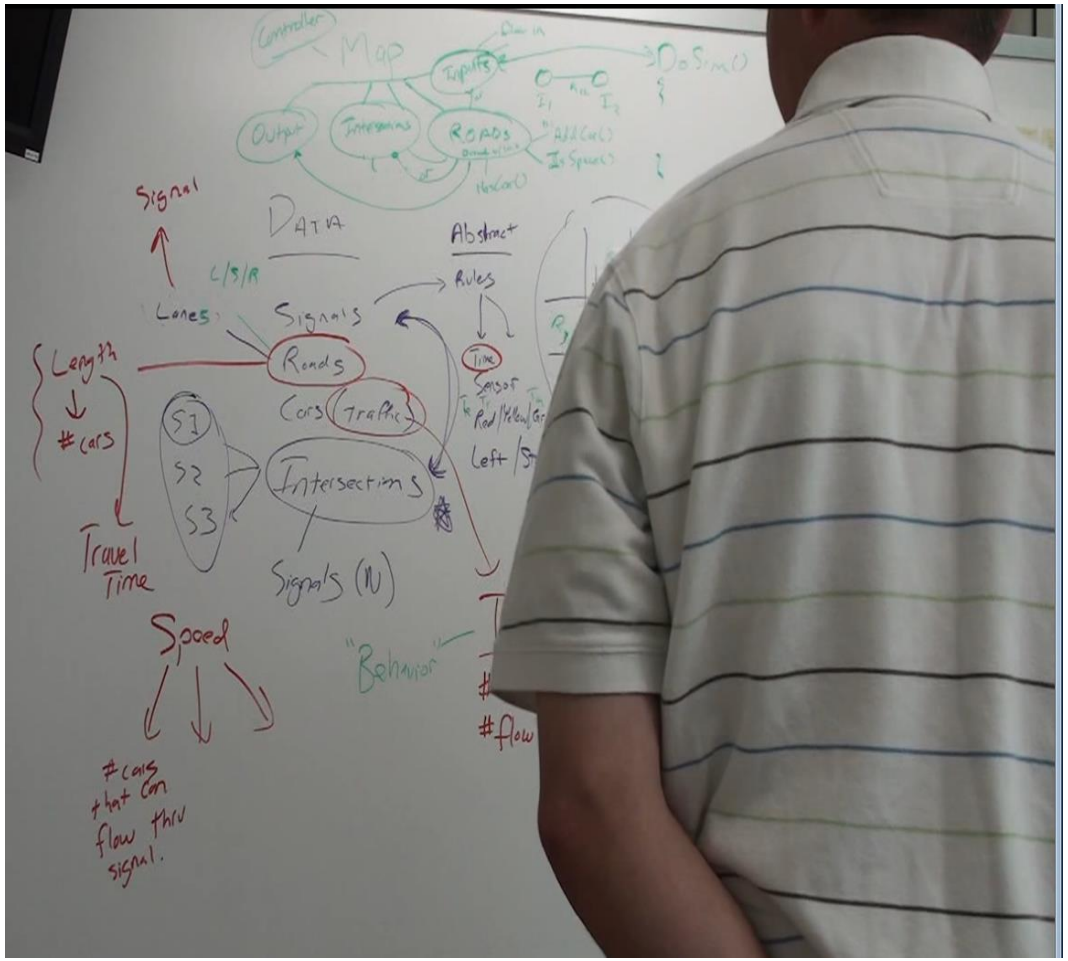
Designers in action [Mirth]

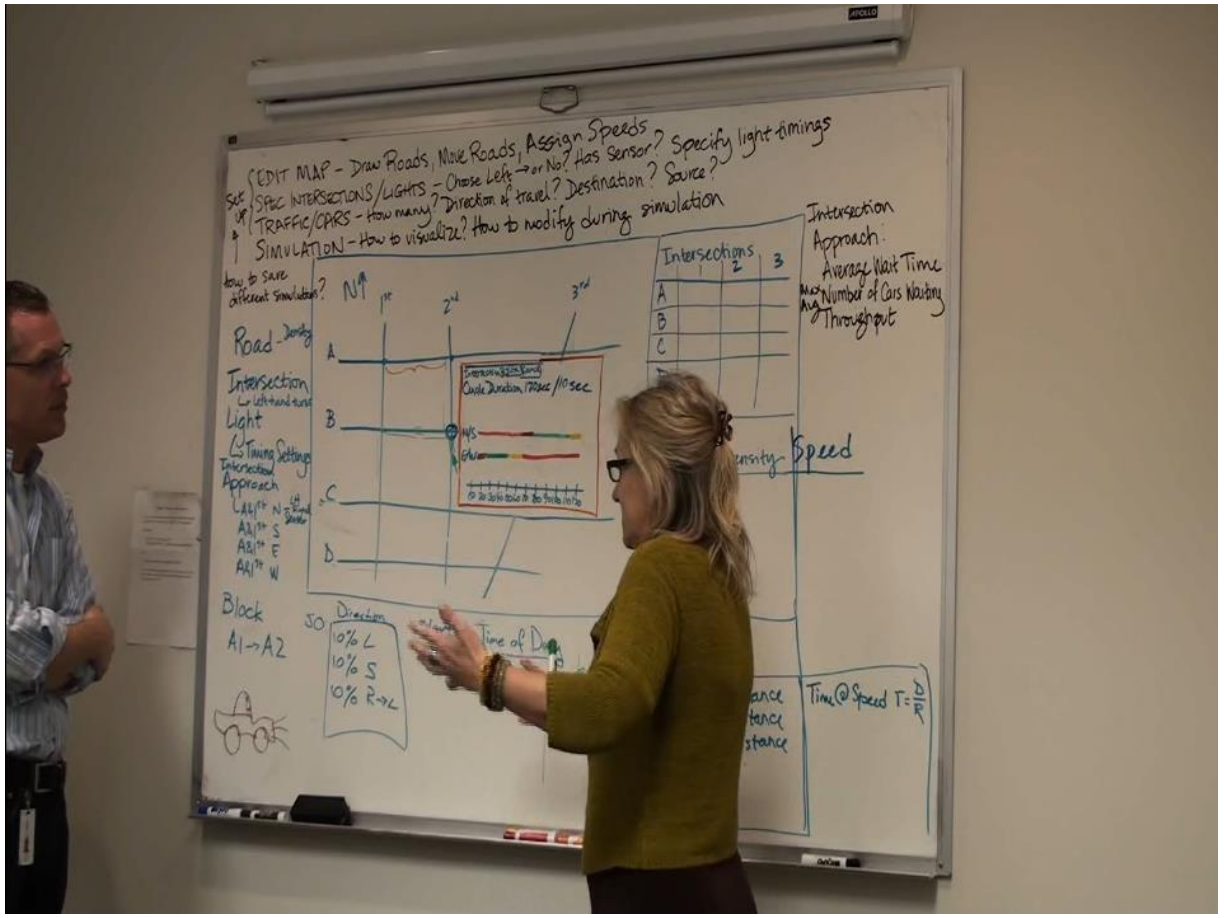




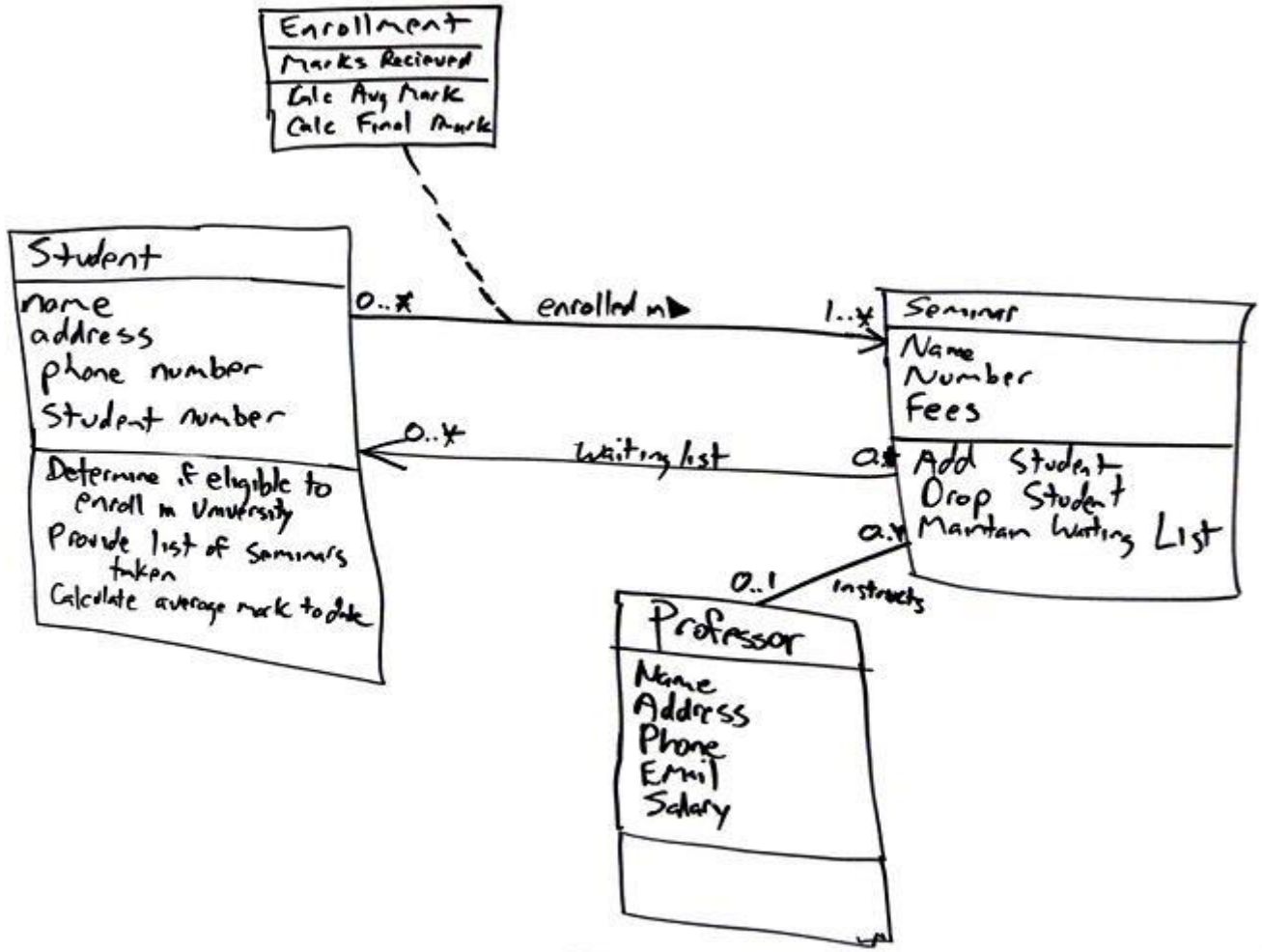
Designers in action [Hortonworks]

Designers in action

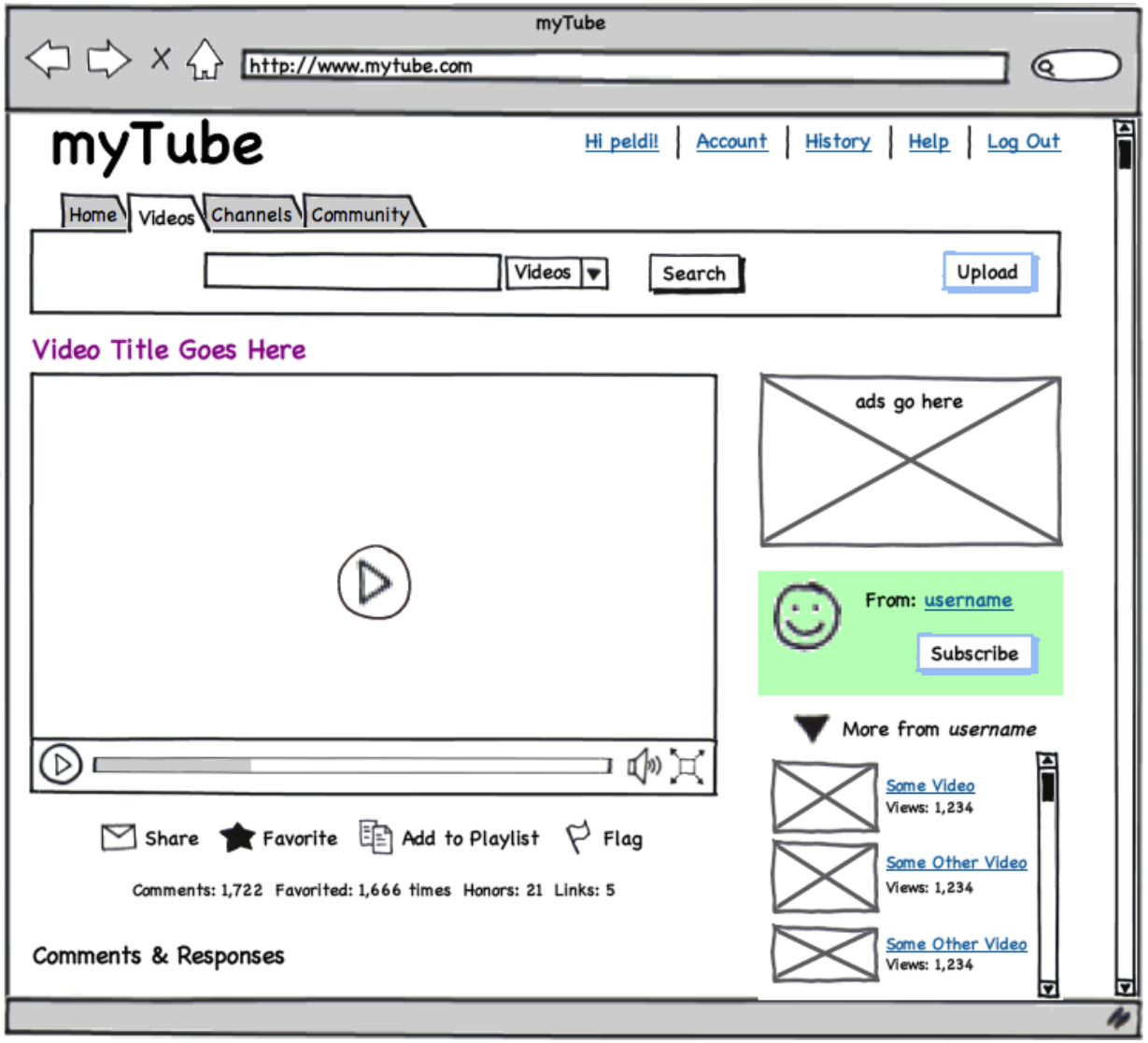




Class diagram

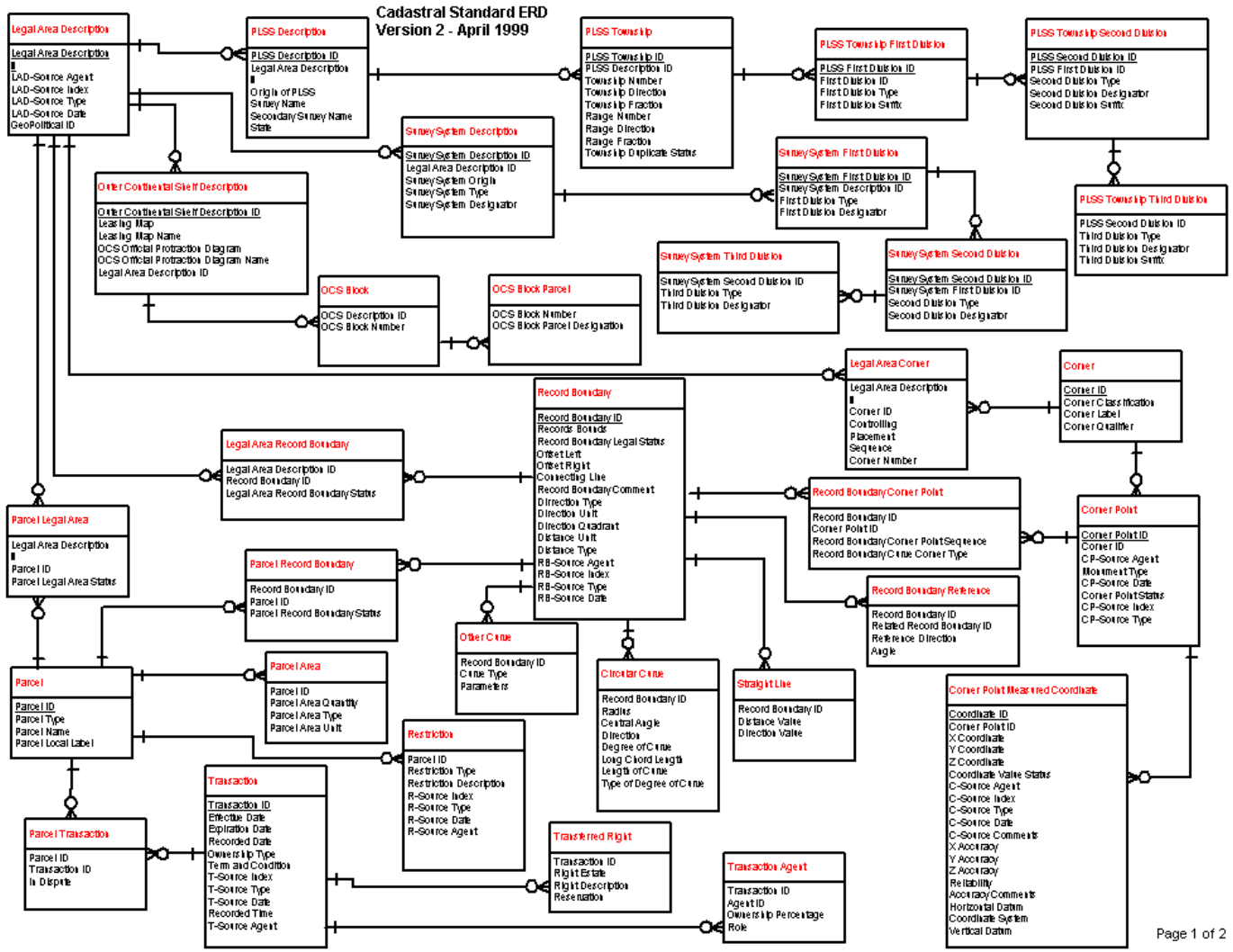


User interface mock-up

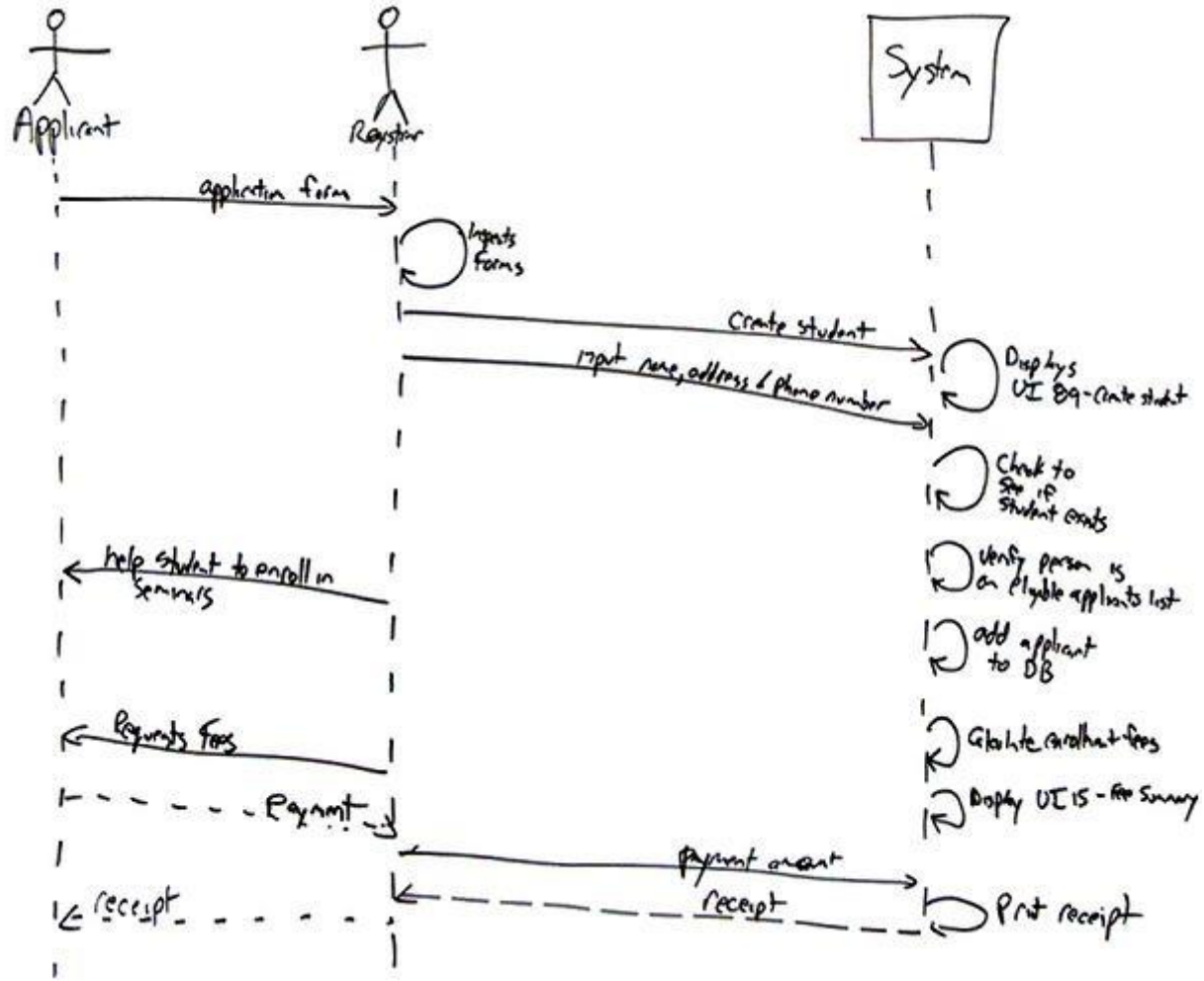


[balsamiq]

Entity relationship diagram



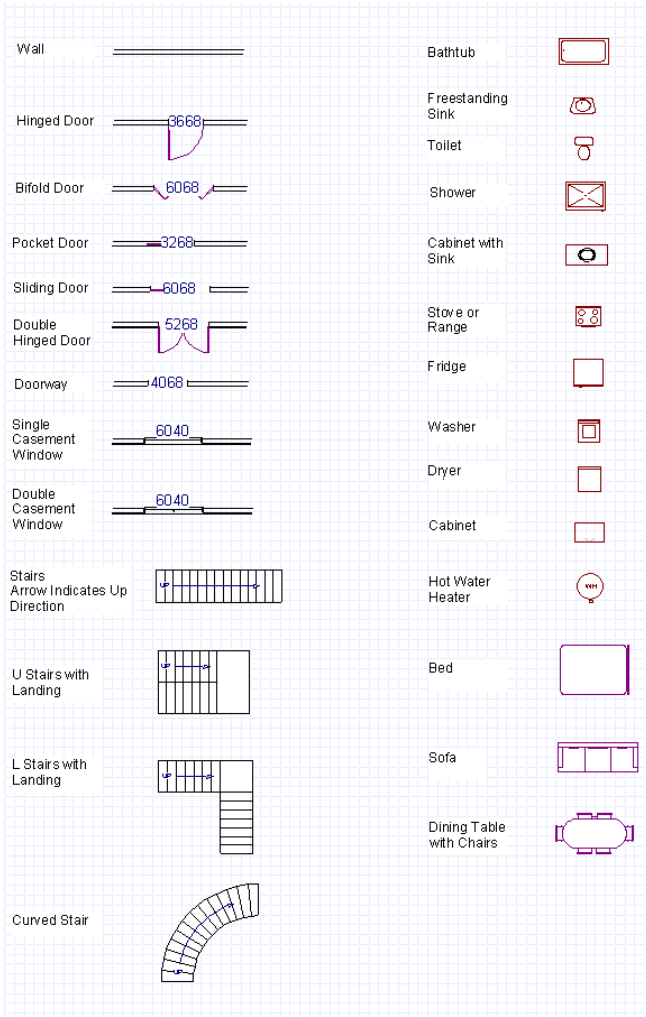
Sequence diagram

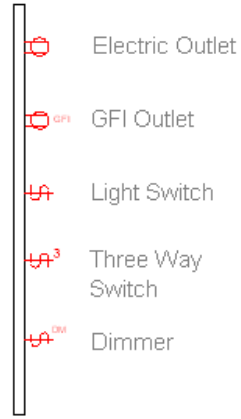


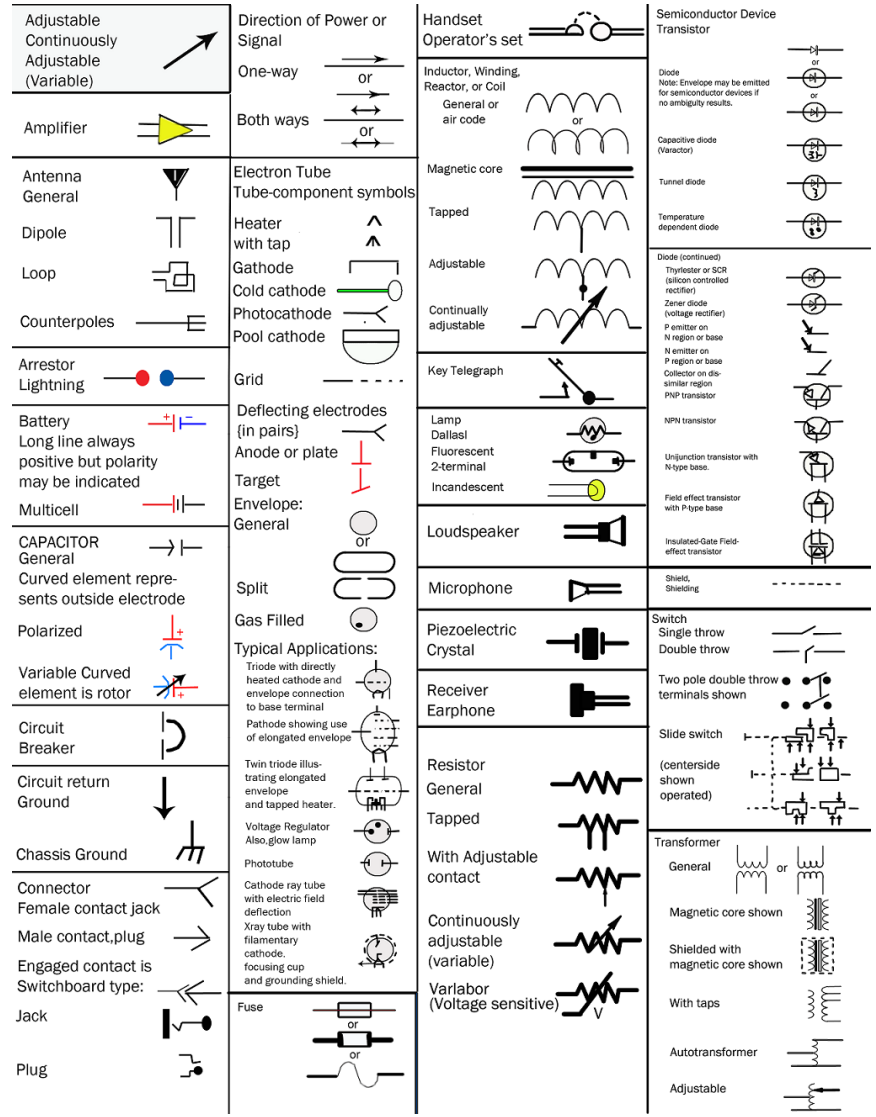
Design notation

- A design notation offers a language for specifying certain aspects of a design artifact
  - textual and/or graphical vocabulary for specifying individual and composite elements
  - rules governing how individual elements can be combined into composite elements
  - implicit and/or explicit semantics for giving meaning
- Each design notation is typically suited for a particular domain and a particular purpose
- Every design notation invariably introduces abstraction

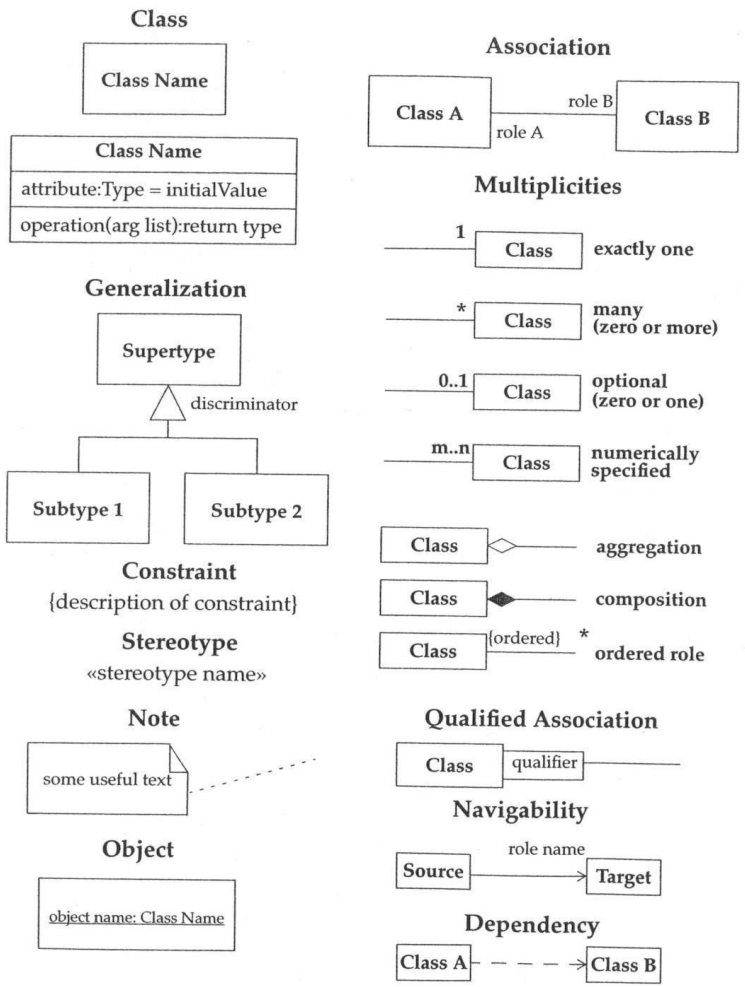
Example notation







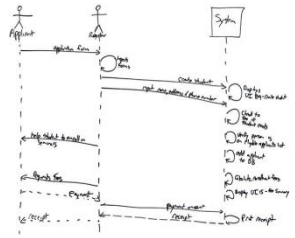
Example notation

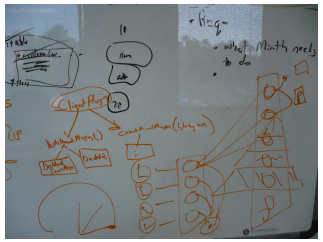


$$
\begin{array}{lcll}
Proc & ::= & STOP & \\
 & | & SKIP & \\
 & | & e \rightarrow Proc & \text{(prefixing)} \\
 & | & Proc \,\Box\, Proc & \text{(external choice)} \\
 & | & Proc \,\sqcap\, Proc & \text{(nondeterministic choice)} \\
 & | & Proc \,|||\, Proc & \text{(interleaving)} \\
 & | & Proc \,|[\{X\}]|\, Proc & \text{(interface parallel)} \\
 & | & Proc \setminus X & \text{(hiding)} \\
 & | & Proc; Proc & \text{(sequential composition)} \\
 & | & \text{if } b \text{ then } Proc \text{ else } Proc & \text{(boolean conditional)} \\
 & | & Proc \,\triangleright\, Proc & \text{(timeout)} \\
 & | & Proc \,\triangle\, Proc & \text{(interrupt)}
\end{array}
$$

Types of notation

$$
\begin{array}{ll}
STOP & \\
SKIP & \\
e \rightarrow Proc & \text{(prefixing)} \\
Proc \,\Box\, Proc & \text{(external choice)} \\
Proc \sqcap Proc & \text{(nondeterministic choice)} \\
Proc \,|||\, Proc & \text{(interleaving)} \\
Proc \,[|\{X\}|]\, Proc & \text{(interface parallel)} \\
Proc \setminus X & \text{(hiding)} \\
Proc; Proc & \text{(sequential composition)} \\
\text{if } b \text{ then } Proc \text{ else } Proc & \text{(boolean conditional)} \\
Proc \triangleright Proc & \text{(timeout)} \\
Proc \,\triangle\, Proc & \text{(interrupt)}
\end{array}
$$





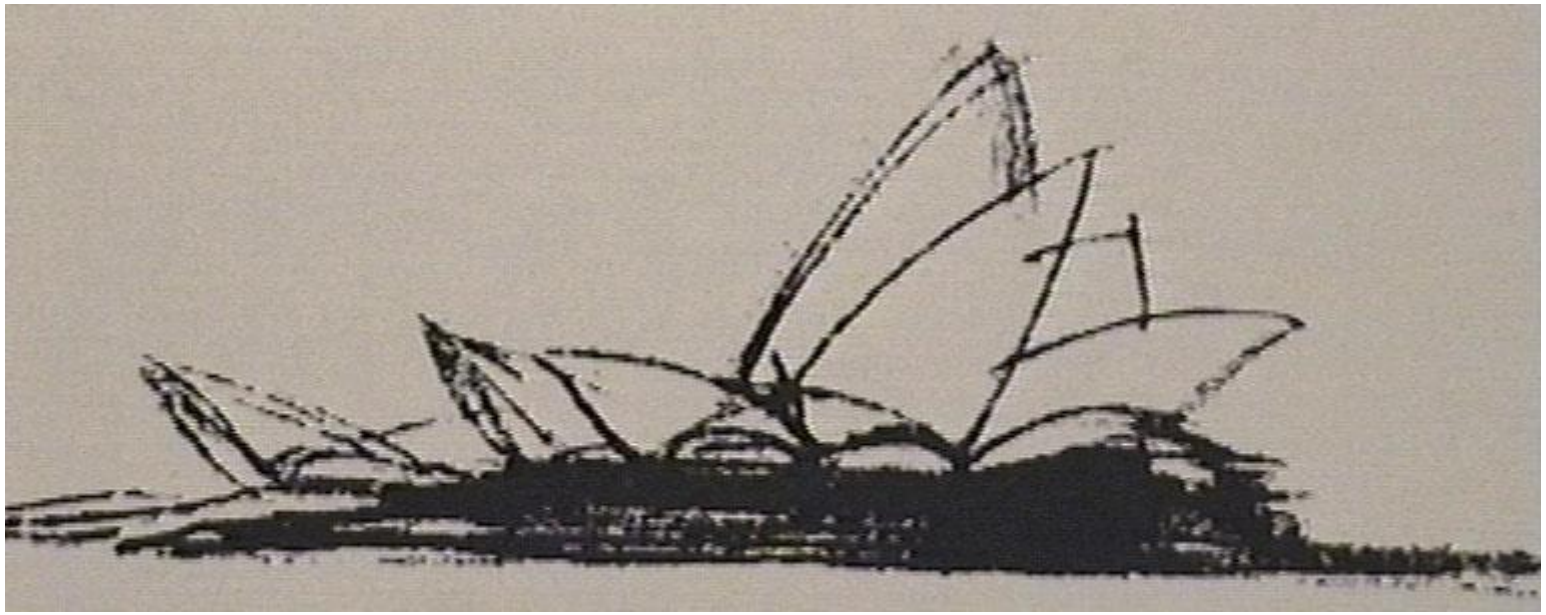
Formal notation

Semi-formal notation

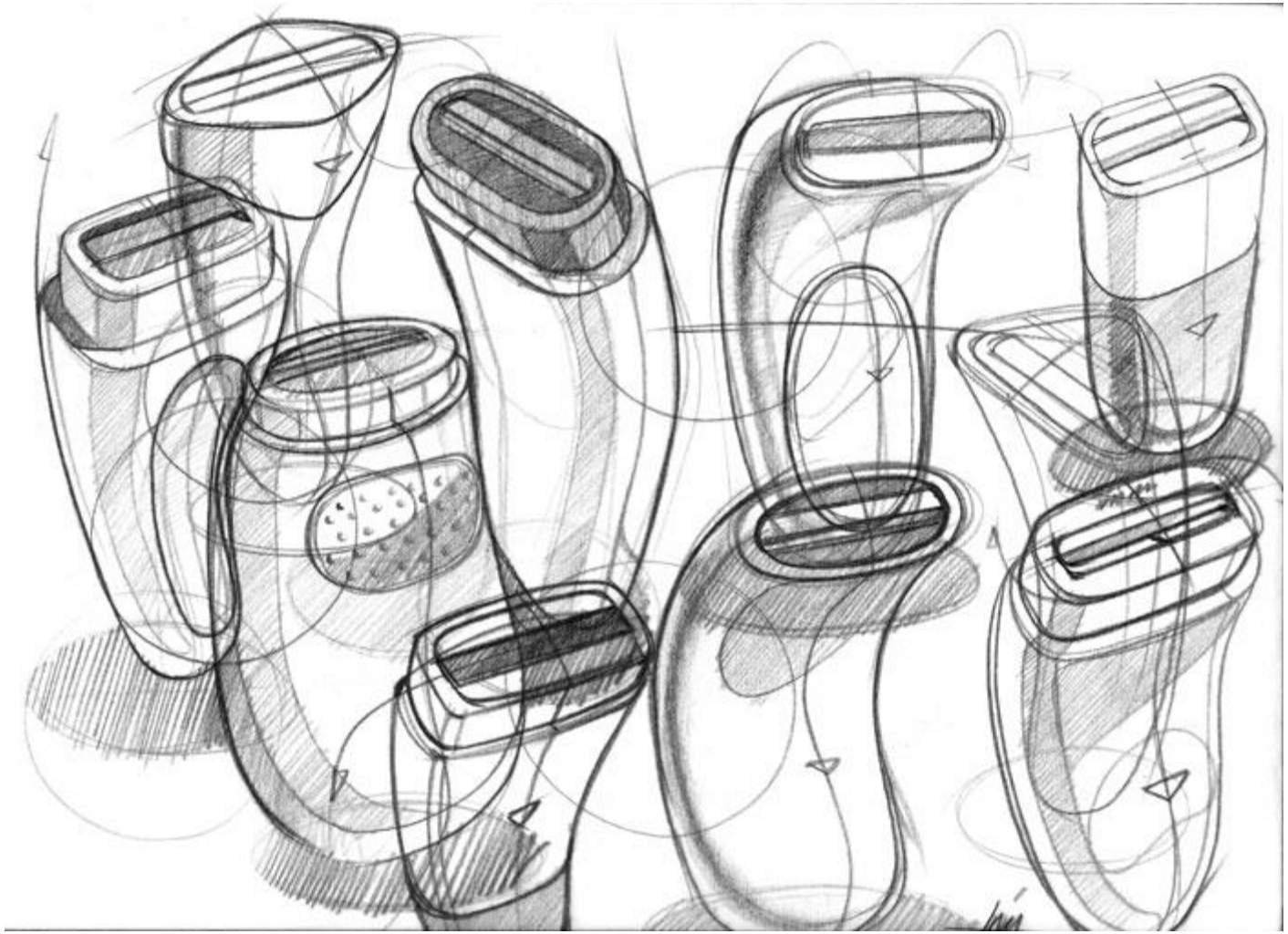
Informal notation

No notation

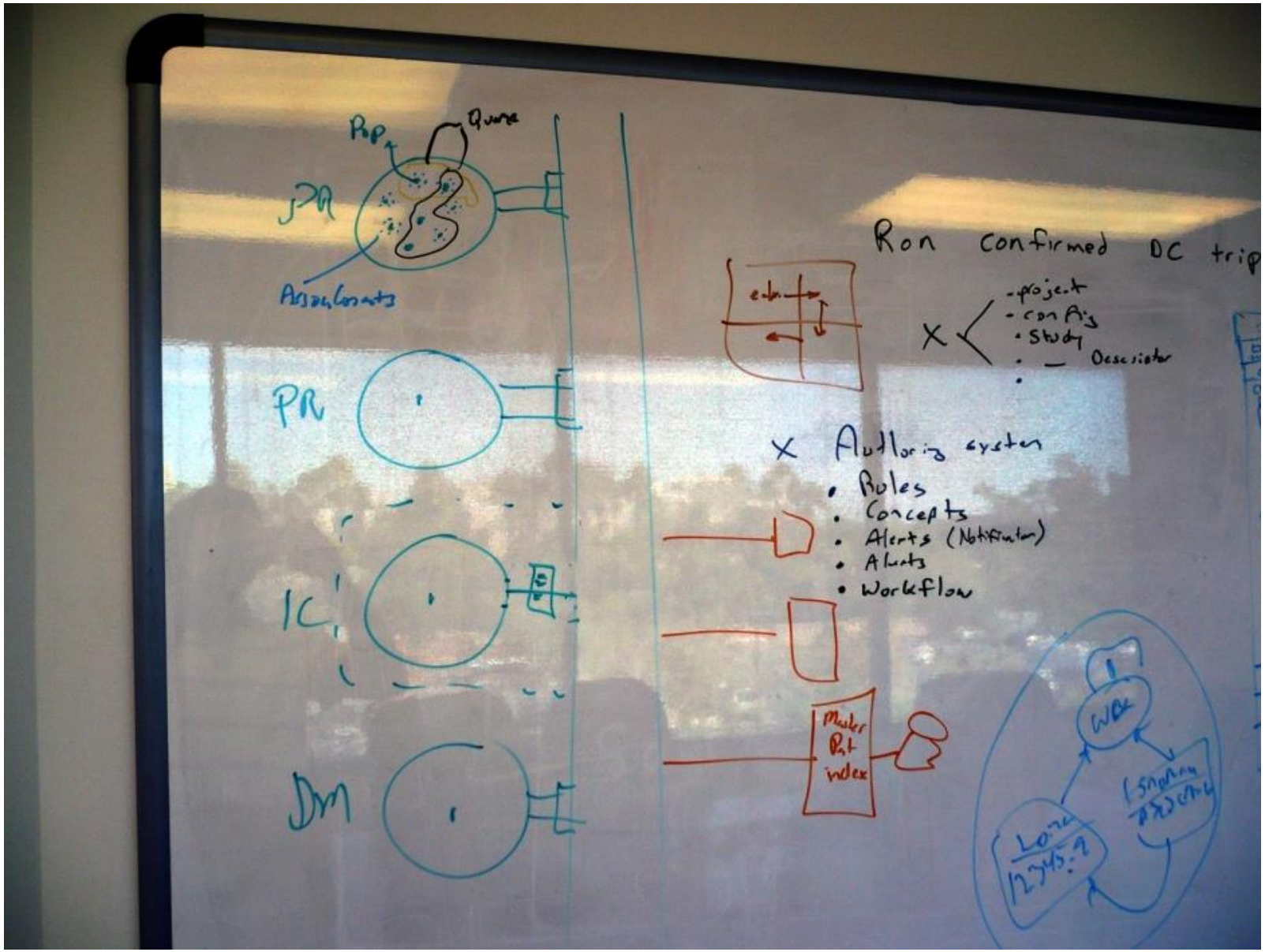
Vocabulary may exist independent of notation



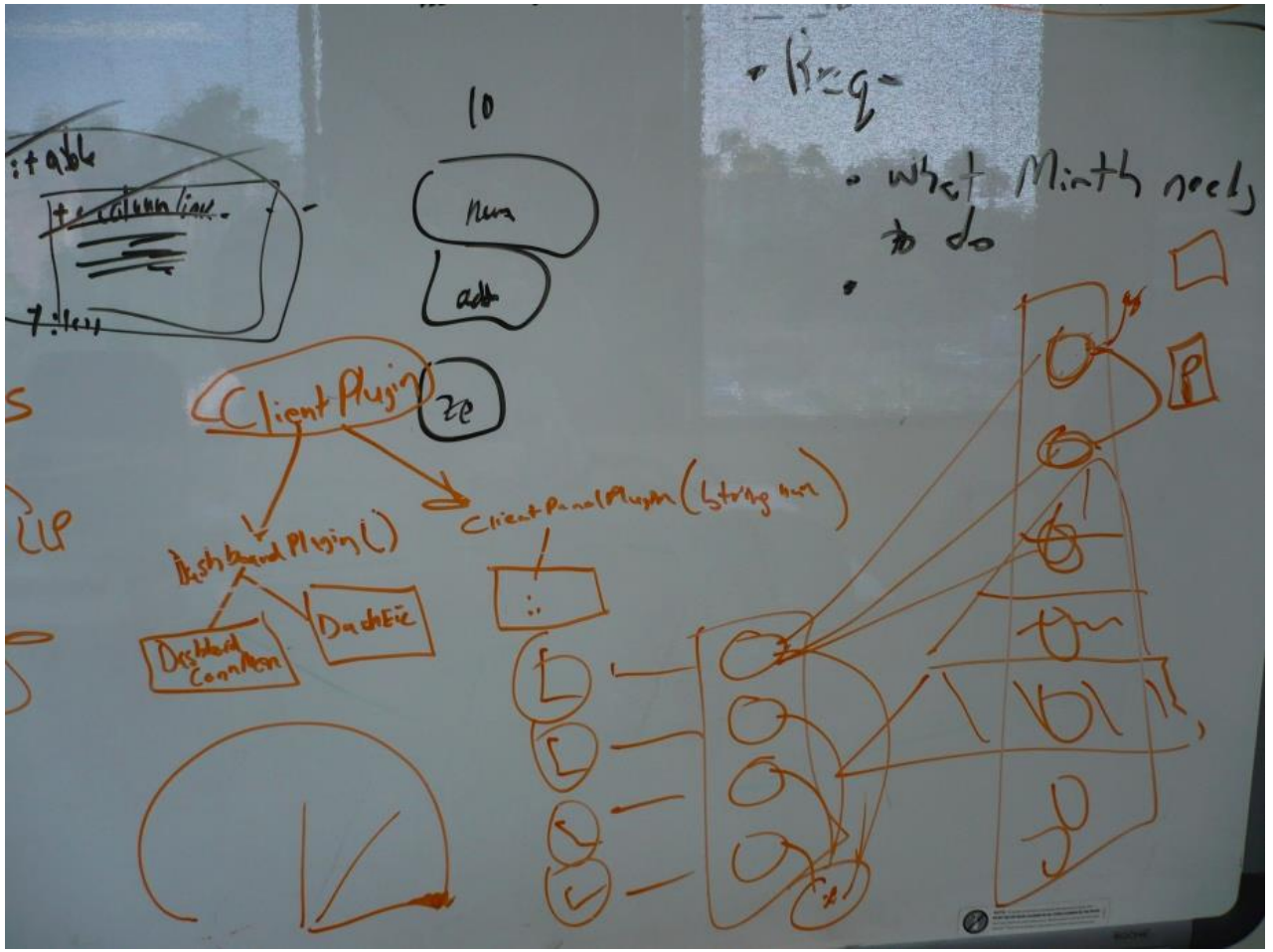
Vocabulary may exist independent of notation



Vocabulary may exist independent of notation



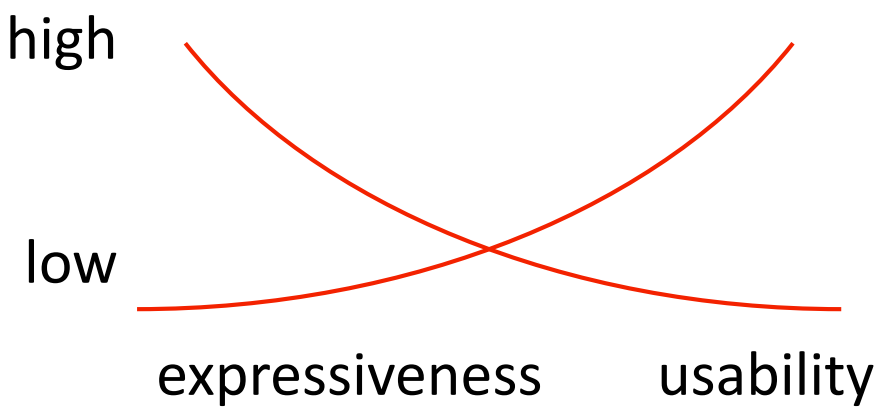
Notation or just a vocabulary?



Considerations in choosing a design notation

- Who is the audience?
- What is the objective?
- What is the timeframe?

Which notation? - Expressiveness versus usability



Cognitive dimensions of notations

Abstraction	types and availability of abstraction mechanisms
Hidden dependencies	important links between entities are not visible
Premature commitment	constraints on the order of doing things
Secondary notation	extra information in means other than formal syntax
Viscosity	resistance to change
Visibility	ability to view components easily
Closeness of mapping	closeness of representation to domain
Consistency	similar semantics are expressed in similar syntactic forms
Diffuseness	verbosity of language
Error-proneness	notation invites mistakes
Hard mental operations	high demand on cognitive resources
Progressive evaluation	work-to-date can be checked at any time
Provisionality	degree of commitment to actions or marks
Role-expressiveness	the purpose of a component is readily inferred

Design tools

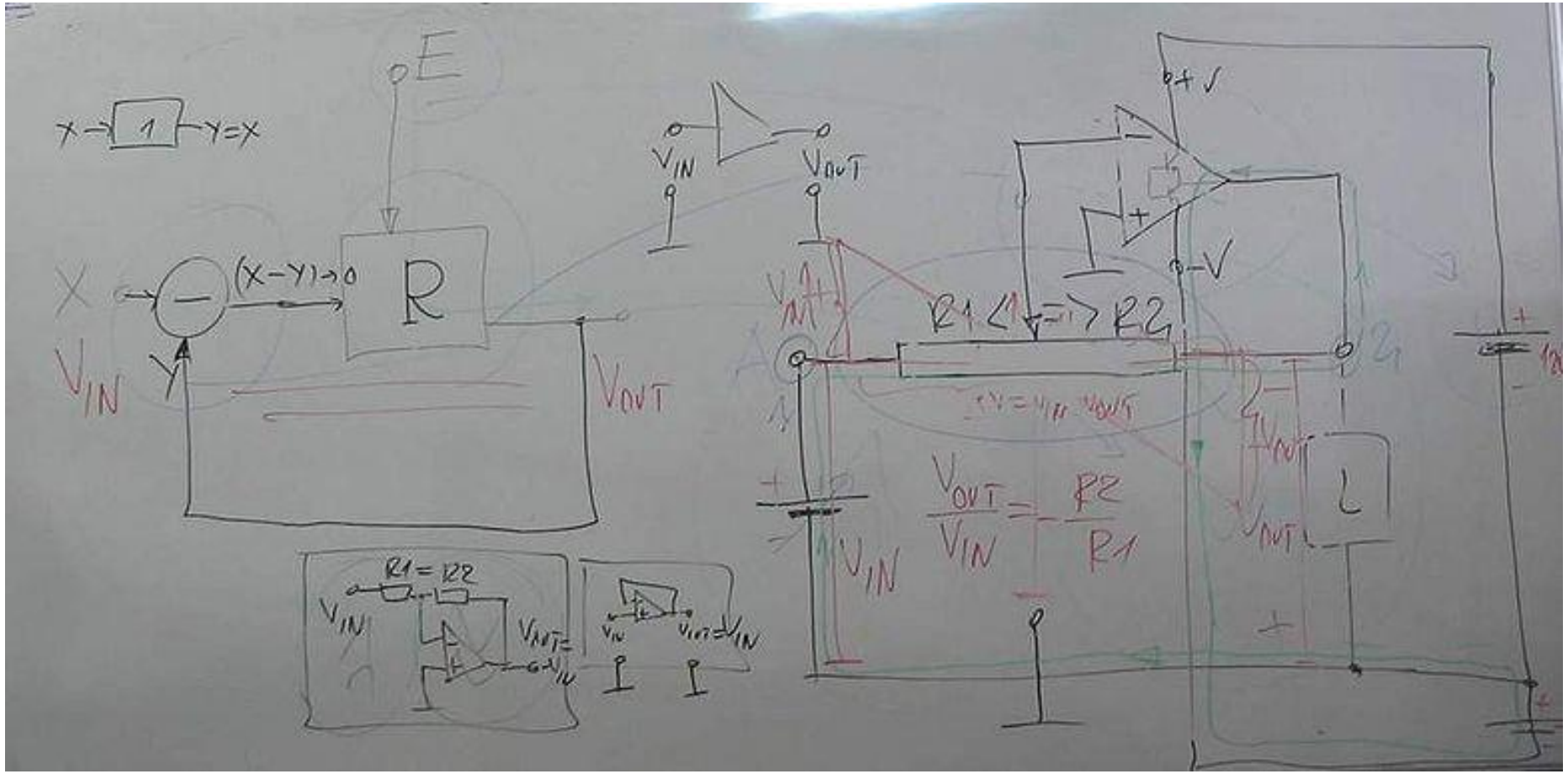
- Design tools assist the designer in creating and interpreting design artifacts

- Ideally, design tools…
  - …decrease the burden of creating design artifacts
  - …increase the ability of interpreting design artifacts
    - static form
    - dynamic behavior

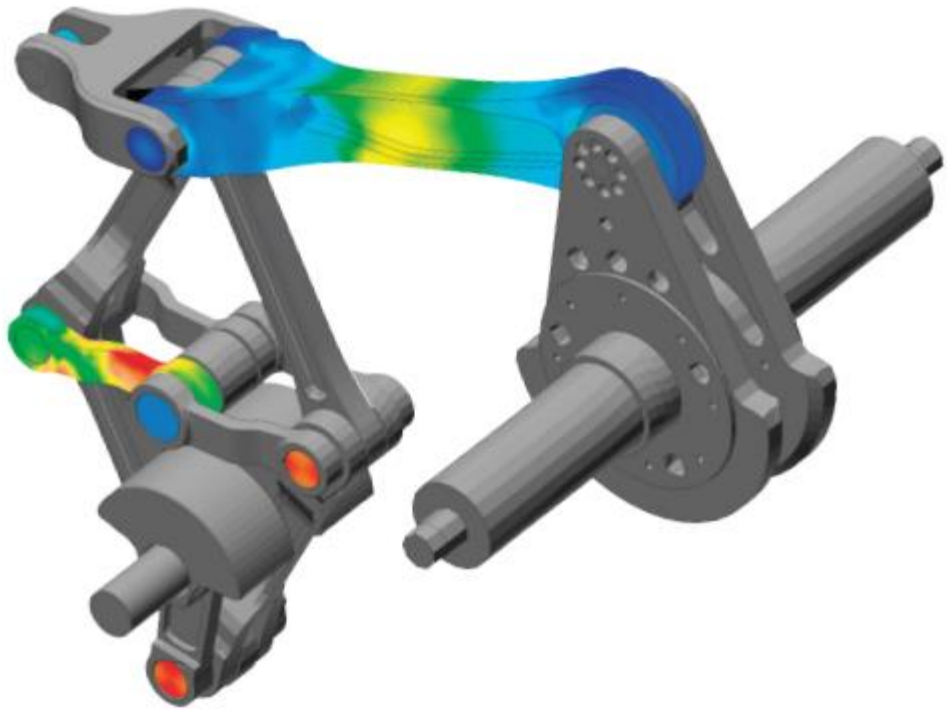
- Design tools may be individual or collaborative in nature

Pen and paper

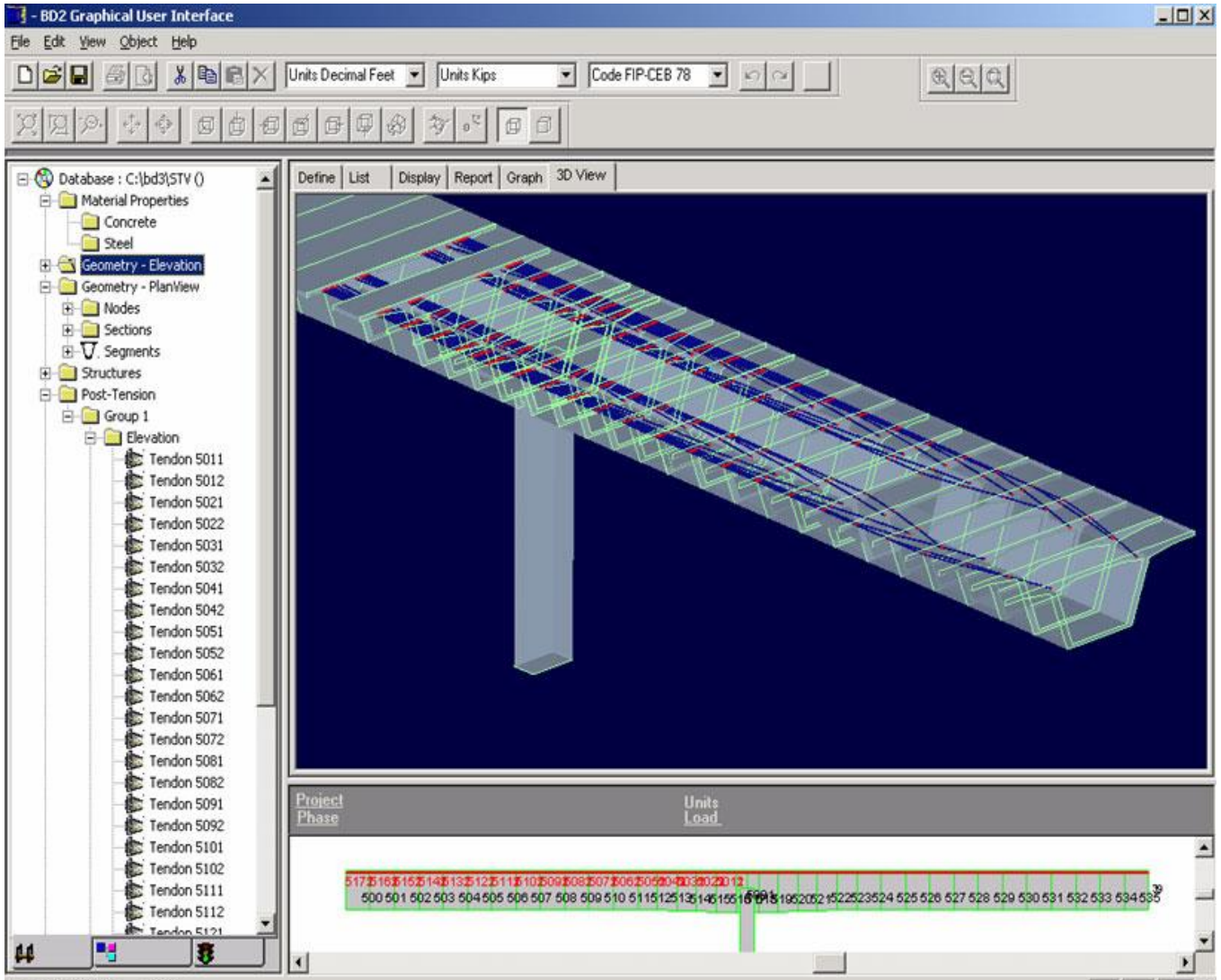
Whiteboard

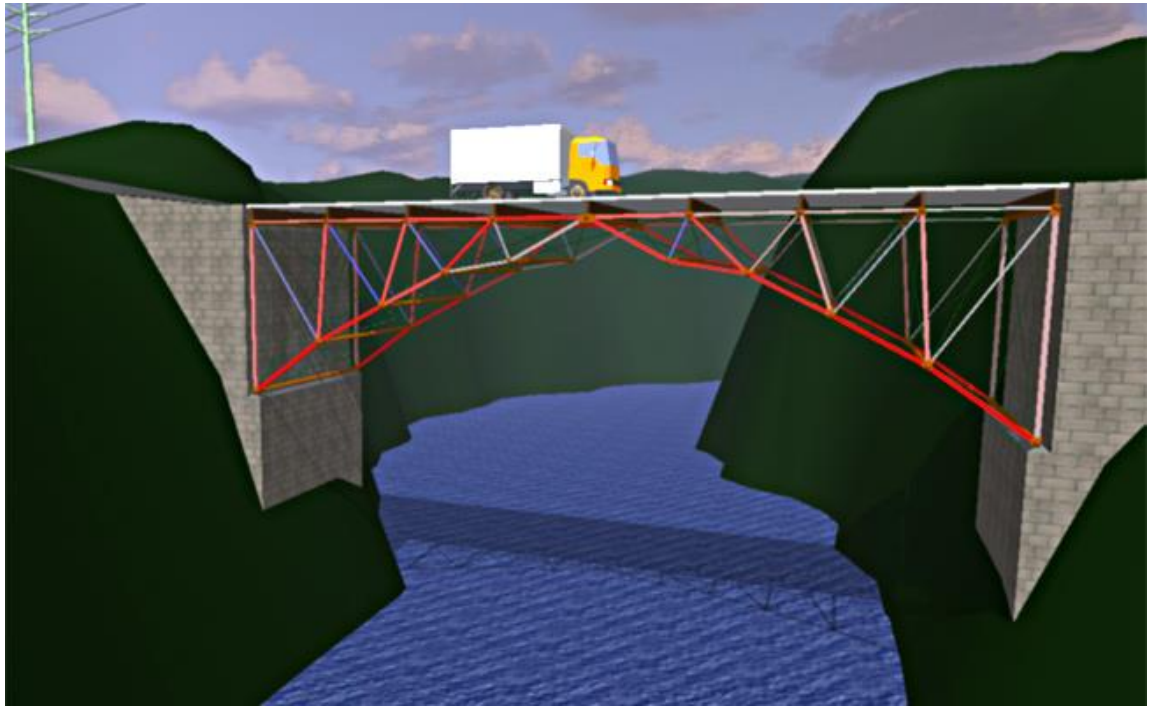


Adobe Illustrator & AutoCAD

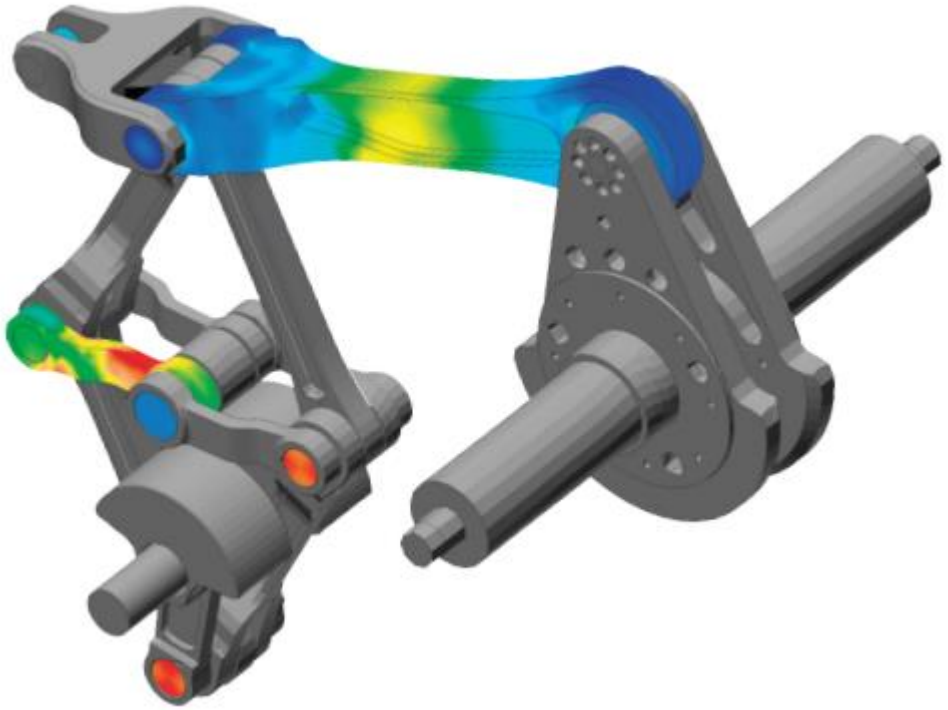


Bridge design

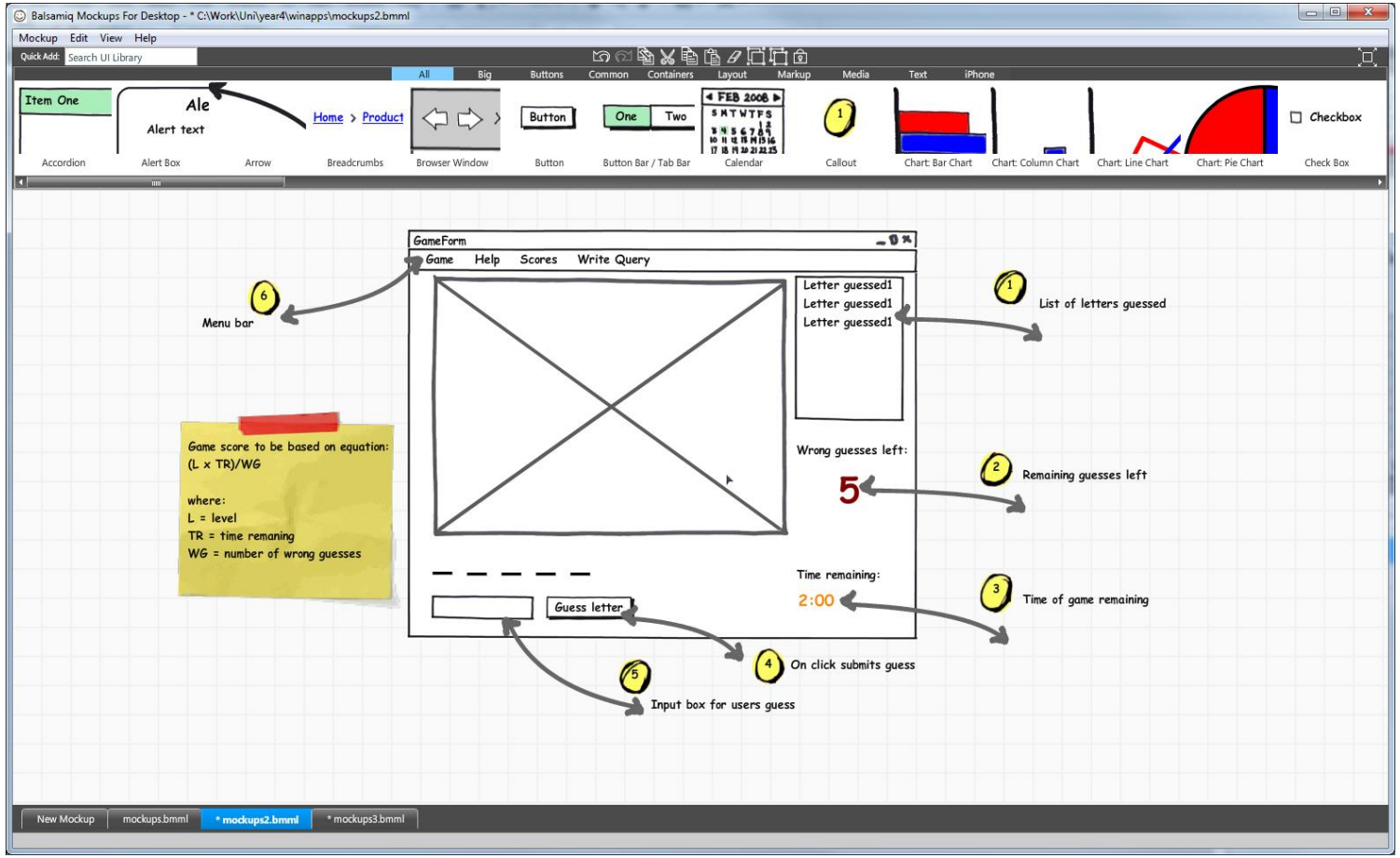




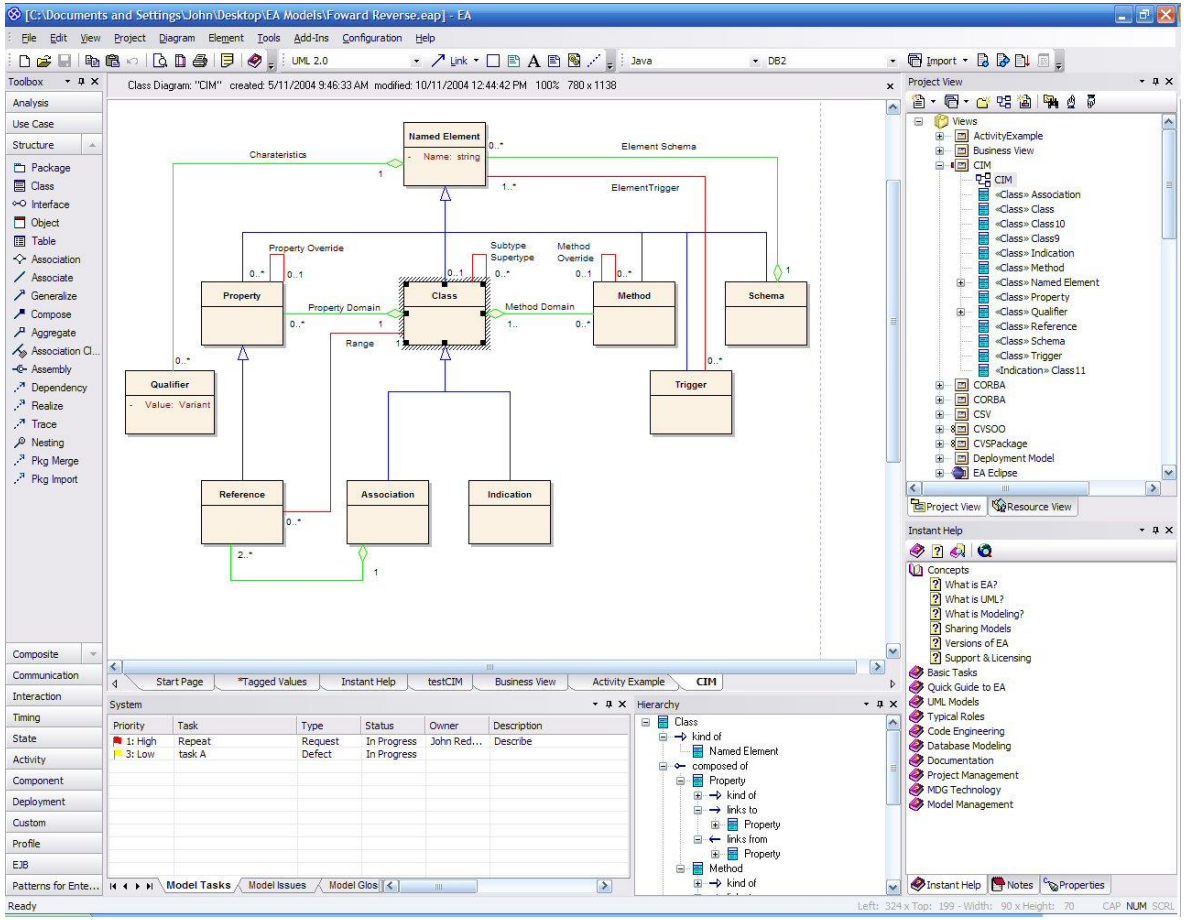
AutoCAD



Balsamiq


Balsamiq Mockups For Desktop - * C:\Work\Uni\year4\winapps\mockups2.bmml
Mockup Edit View Help
Quick Add: Search UI Library
All Big Buttons Common Containers Layout Markup Media Text iPhone
Item One
Ale
Alert text
Home > Product
Button
One Two
FEB 2008
Checkbox
Accordion
Alert Box
Arrow
Breadcrumbs
Browser Window
Button
Button Bar / Tab Bar
Calendar
Callout
Chart: Bar Chart
Chart: Column Chart
Chart: Line Chart
Chart: Pie Chart
Check Box
GameForm
Game Help Scores Write Query
Menu bar
Letter guessed1
Letter guessed1
Letter guessed1
List of letters guessed
Game score to be based on equation:
(L x TR)/WG
where:
L = level
TR = time remaning
WG = number of wrong guesses
Wrong guesses left:
5
Remaining guesses left
Time remaining:
2:00
Time of game remaining
Guess letter
On click submits guess
Input box for users guess
New Mockup
mockups.bmml
* mockups2.bmml
* mockups3.bmml

Enterprise Architect



Does design work?

Design failures

YEAR	COMPANY	OUTCOME (COSTS IN US $)
2005	Hudson Bay Co. [Canada]	Problems with inventory system contribute to $33.3 million* loss.
2004–05	UK Inland Revenue	Software errors contribute to $3.45 billion* tax-credit overpayment.
2004	Avis Europe PLC [UK]	Enterprise resource planning (ERP) system canceled after $54.5 million† is spent.
2004	Ford Motor Co.	Purchasing system abandoned after deployment costing approximately $400 million.
2004	J Sainsbury PLC [UK]	Supply-chain management system abandoned after deployment costing $527 million.†
2004	Hewlett-Packard Co.	Problems with ERP system contribute to $160 million loss.
2003–04	AT&T Wireless	Customer relations management (CRM) upgrade problems lead to revenue loss of $100 million.
2002	McDonald's Corp.	The Innovate information-purchasing system canceled after $170 million is spent.
2002	Sydney Water Corp. [Australia]	Billing system canceled after $33.2 million† is spent.
2002	CIGNA Corp.	Problems with CRM system contribute to $445 million loss.
2001	Nike Inc.	Problems with supply-chain management system contribute to $100 million loss.
2001	Kmart Corp.	Supply-chain management system canceled after $130 million is spent.
2000	Washington, D.C.	City payroll system abandoned after deployment costing $25 million.

Two fundamental challenges

- The nature of software
- The nature of people

Nature of software (Brooks)

- Complexity
  - software is among the most complex people-made artifacts
- Conformity
  - software has no laws of nature that simplify its existence; rather, it lives in a world of designed artifacts to which it much conform
- Changeability
  - software is subject to continuous pressure to change
- Invisibility
  - because the reality of software is not embedded into space, it is inherently unvisualizable

Nature of people

- Diversity
  - people differ in how they experience the world
- Indiscernibility
  - experiences are distinctly mental in nature, with tangible reactions and signs not always matching their actual experience
- Familiarity
  - people tend to be risk averse, sticking to role, organizational, and societal norms and values
- Volatility
  - with every new exposure, people reinterpret and modify their opinions and expectations

Why software design is challenging?

- No physical artifacts
- Lack of clarity
- Mind boggling complexity
- Drastic technology changes
- Failures often but not tolerable
- Change is expected rapidly
- ….

- Explicit Versus Implicit
- Tangible Versus Intangible
- Manageable Complexity Versus Unmanageable Complexity
- Changeable Environment Versus
- Unchangeable Environment
- No Major Changes Versus Major Changes

Seven difficulties every software designer faces

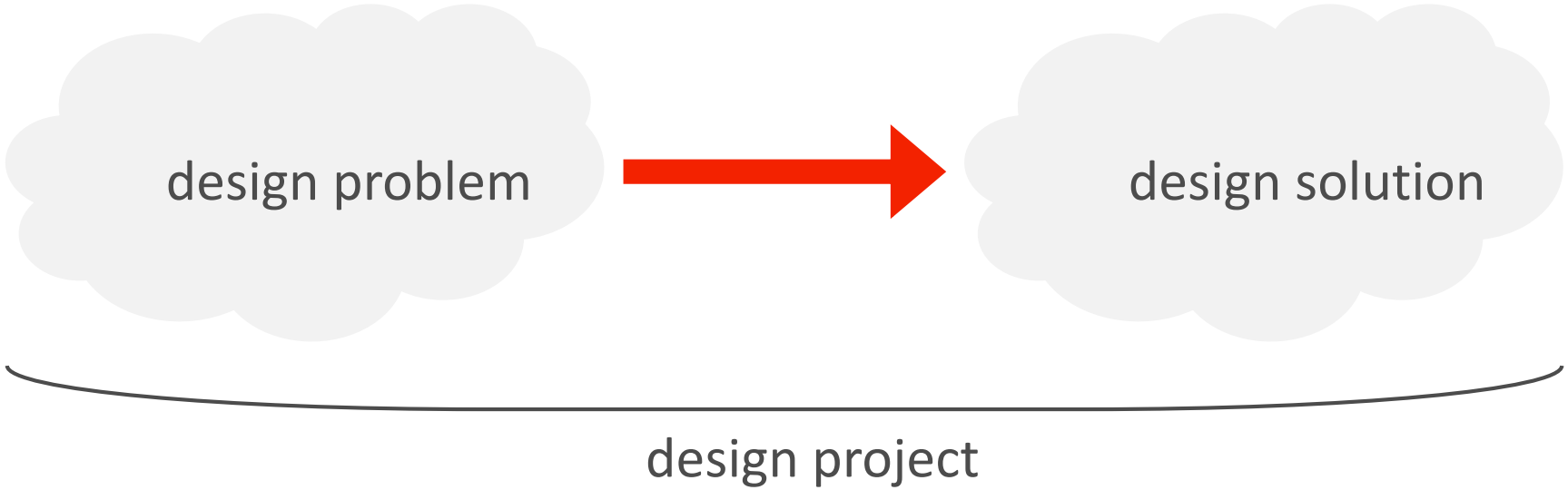
- Prediction: people
- Tradeoffs:: quantify
- Change
- Bias
- Longevity:legacy
- Quality, cost, and effort
- Source code as a design notation

Software design is a wicked problem

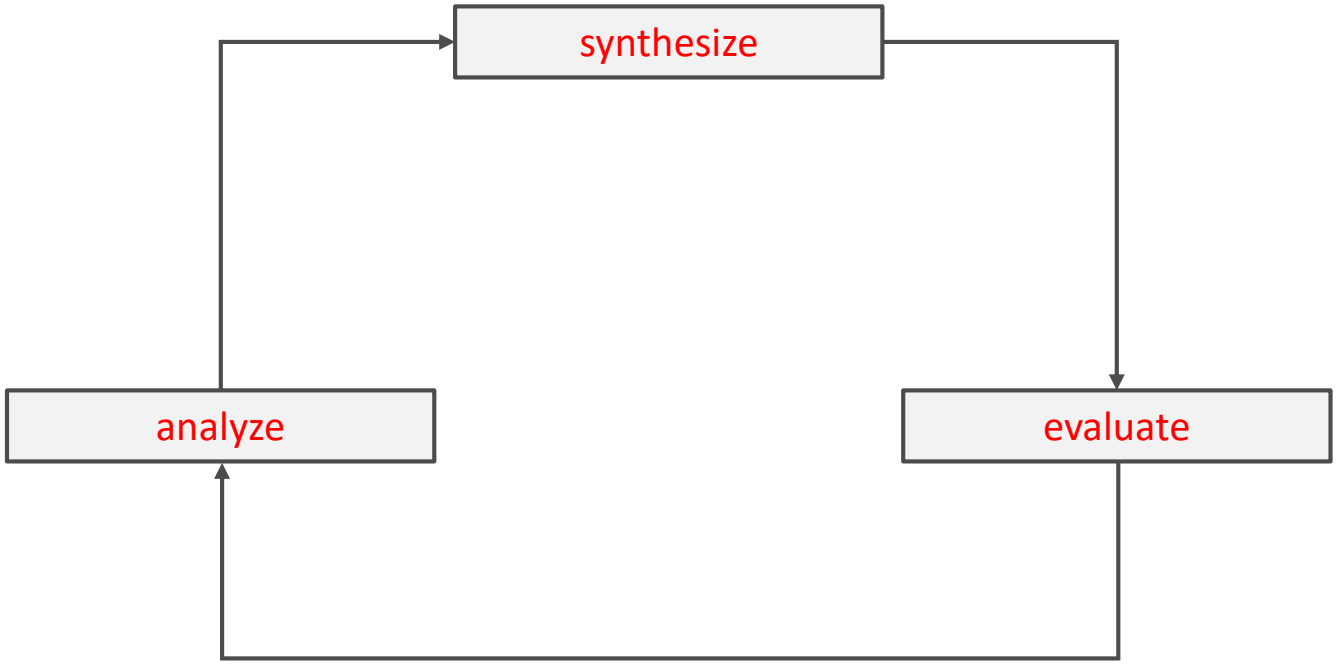
- The problem is not understood until after the formulation of a solution
- Wicked problems have no stopping rule
- Solutions to wicked problems are not right or wrong
- Every wicked problem is essentially novel and unique
- Every solution to a wicked problem is a “one shot operation”
- Wicked problems have no given alternative solutions

Worse, software design is a ‘hard’ wicked problem

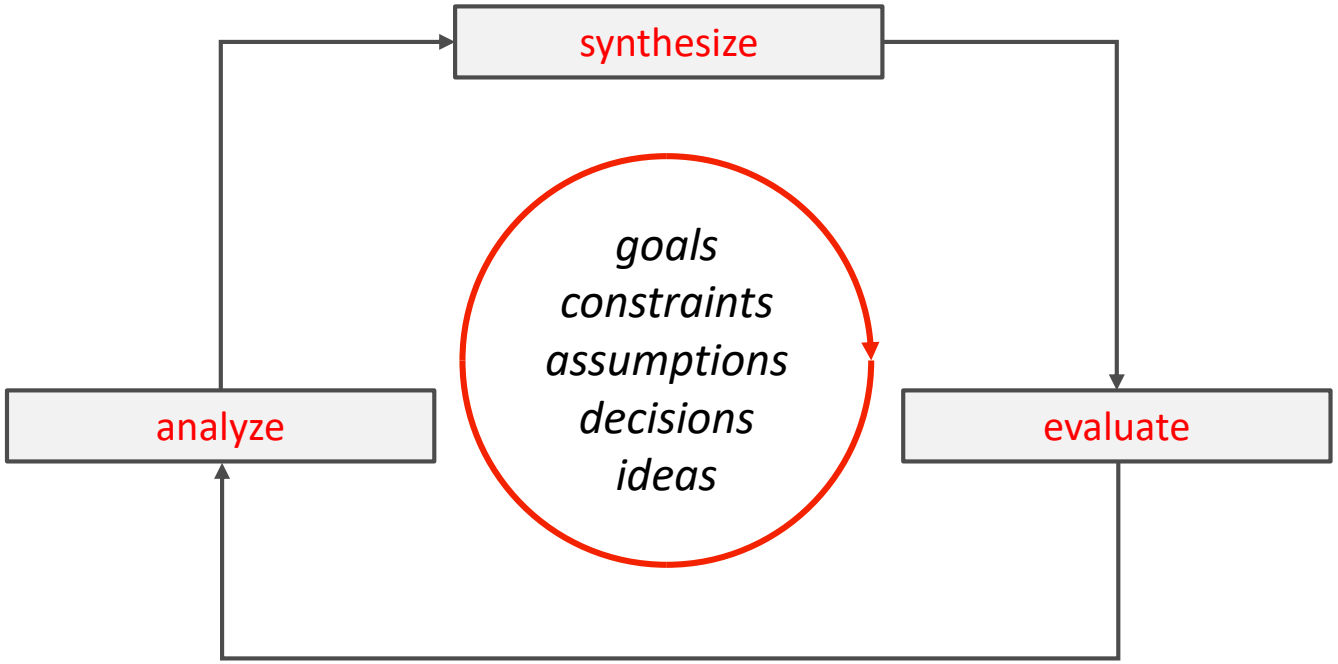
Design project



Design cycle



Design cycle



Software design methods

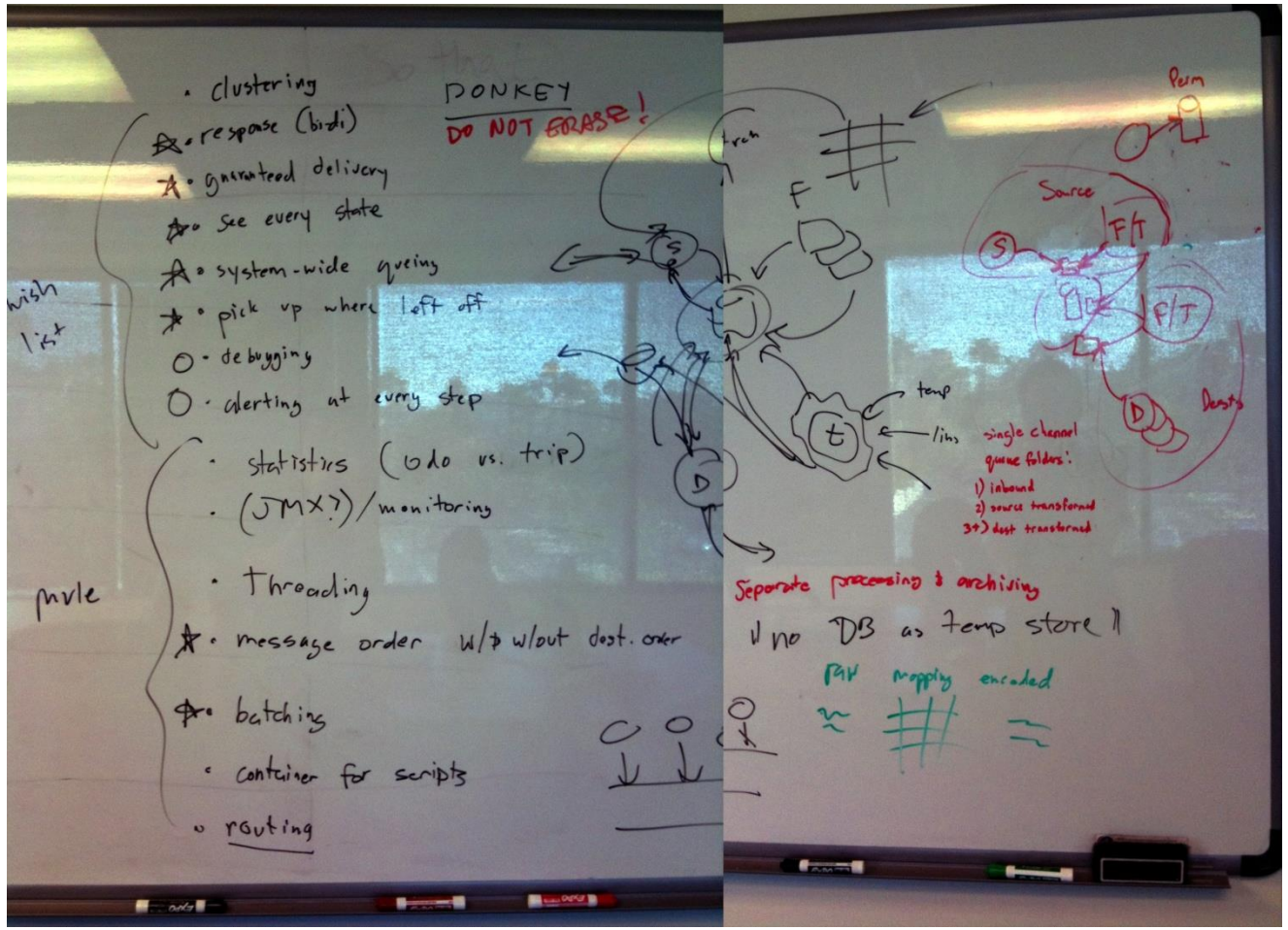
	Application design	Interaction design	Architecture design	Implementation design
Analysis	• competitive testing • contextual inquiry • feature comparison • stakeholder analysis • task analysis	• critical incident technique • interaction logging • personas • scenarios	• framework assessment • model-driven engineering • quality-function-deployment • reverse engineering • world modeling	• release planning • summarization • test-driven design • visualization
Synthesis	• affinity diagramming • concept mapping • mind mapping • morphological chart	• design/making • participatory design • prototyping • storyboarding	• architectural styles • generative programming • component reuse • decomposition	• pair programming • refactoring • search • software patterns
Evaluation	• requirements review • role playing • wizard of oz	• cognitive walkthrough • evaluative research • heuristic evaluation • think-aloud protocol	• formal verification • simulation • weighted objectives	• correctness proofs • inspections/reviews • parallel deployment • testing

What do expert designers do?

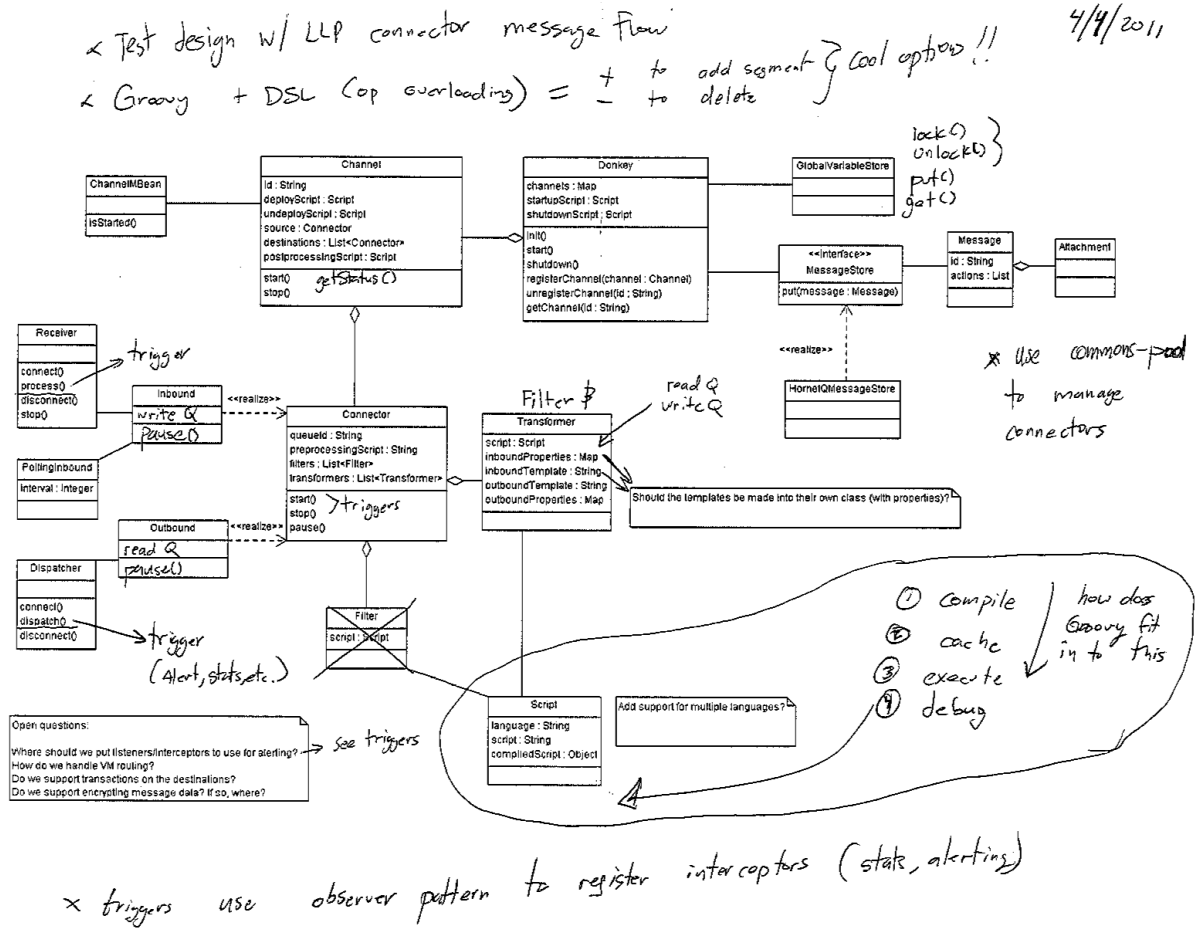
- Experts prefer solutions that they know work
- Experts look around
- Experts take inspiration from wherever they can
- Experts use analogy
- Experts use design methods (selectively)
- Experts network

A Design Journey

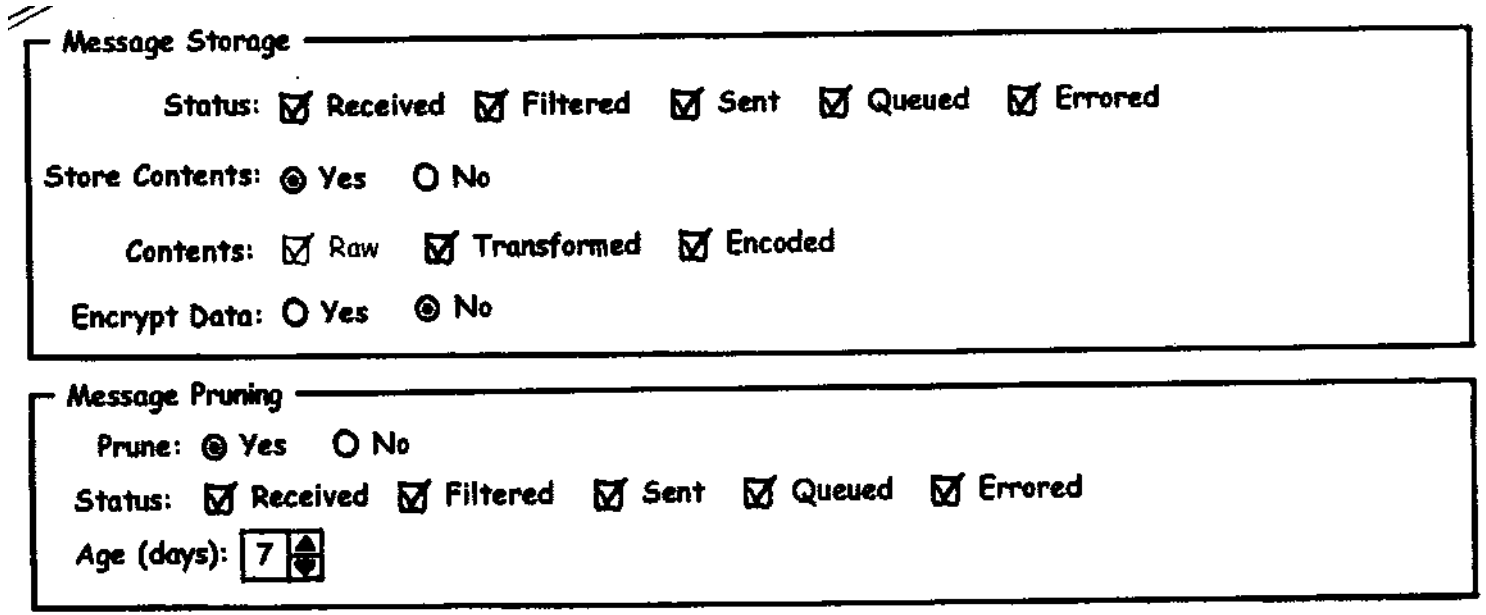
March 25, 2011



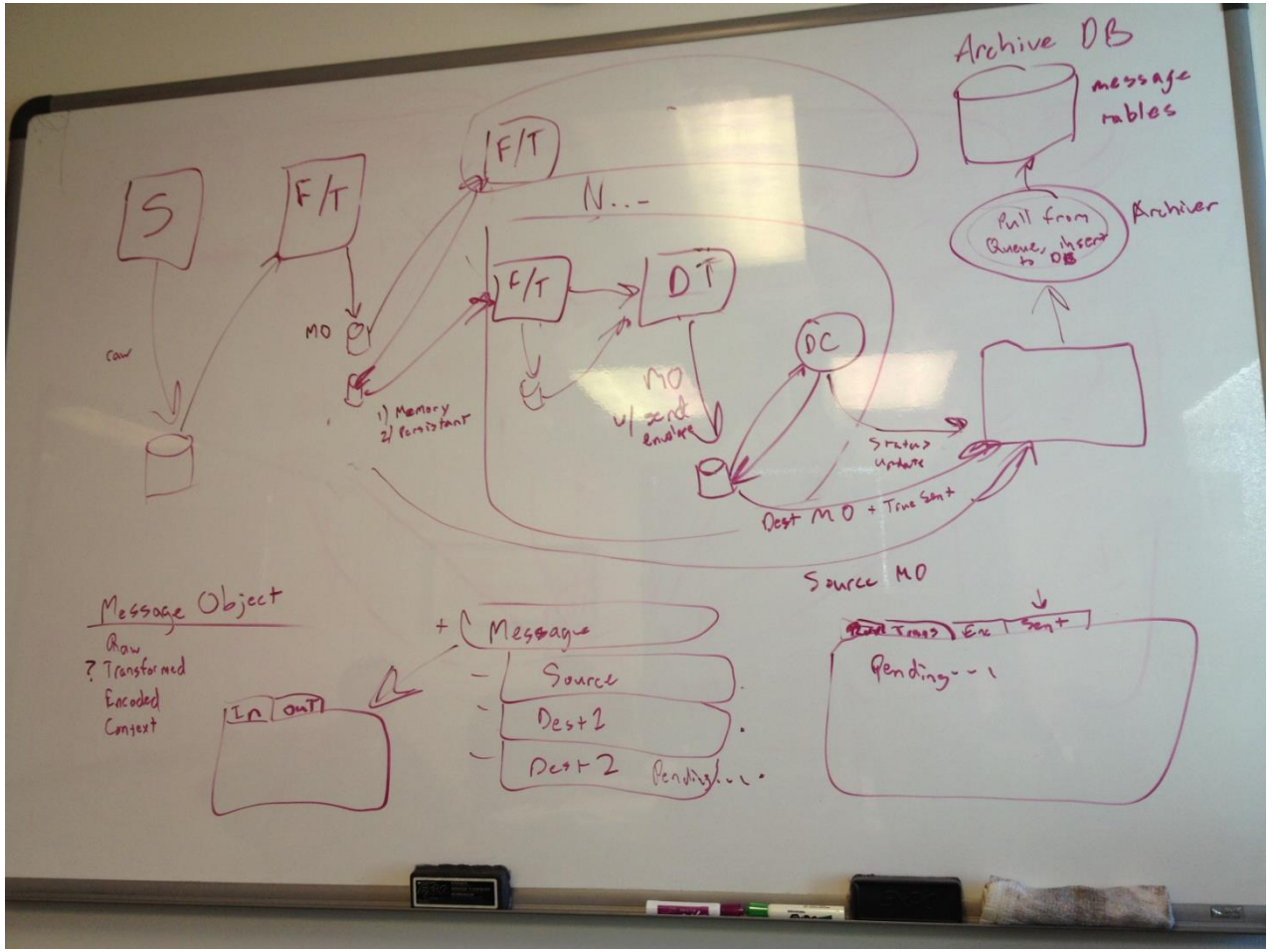
April 4, 2011


4/4/2011
Test design w/ LLP connector message flow
Groovy + DSL (op overloading) = + to add segment } Cool options!!
- to delete
lock()
unlock()
put()
get()
ChannelMBean
isStarted()
Channel
id : String
deployScript : Script
undeployScript : Script
source : Connector
destinations : List<Connector>
postprocessingScript : Script
start()
stop()
getStatus()
Donkey
channels : Map
startupScript : Script
shutdownScript : Script
init()
start()
shutdown()
registerChannel(channel : Channel)
unregisterChannel(id : String)
getChannel(id : String)
GlobalVariableStore
<<Interface>>
MessageStore
put(message : Message)
<<realize>>
HornetQMessageStore
Message
id : String
actions : List
Attachment
Receiver
connect()
process()
disconnect()
stop()
trigger
Inbound
write Q
pause()
<<realize>>
Connector
queueId : String
preprocessingScript : String
filters : List<Filter>
transformers : List<Transformer>
start()
stop()
pause()
triggers
Filter
Transformer
script : Script
inboundProperties : Map
inboundTemplate : String
outboundTemplate : String
outboundProperties : Map
read Q
write Q
Should the templates be made into their own class (with properties)?
use commons-pool to manage connectors
PollingInbound
interval : Integer
Outbound
read Q
pause()
<<realize>>
Dispatcher
connect()
dispatch()
disconnect()
trigger (Alert, stats, etc.)
Filter
script : Script
Script
language : String
script : String
compiledScript : Object
Add support for multiple languages?
1 compile
2 cache
3 execute
4 debug
how does Groovy fit in to this
Open questions:
Where should we put listeners/interceptors to use for alerting?
How do we handle VM routing?
Do we support transactions on the destinations?
Do we support encrypting message data? If so, where?
see triggers
triggers use observer pattern to register interceptors (stats, alerting)

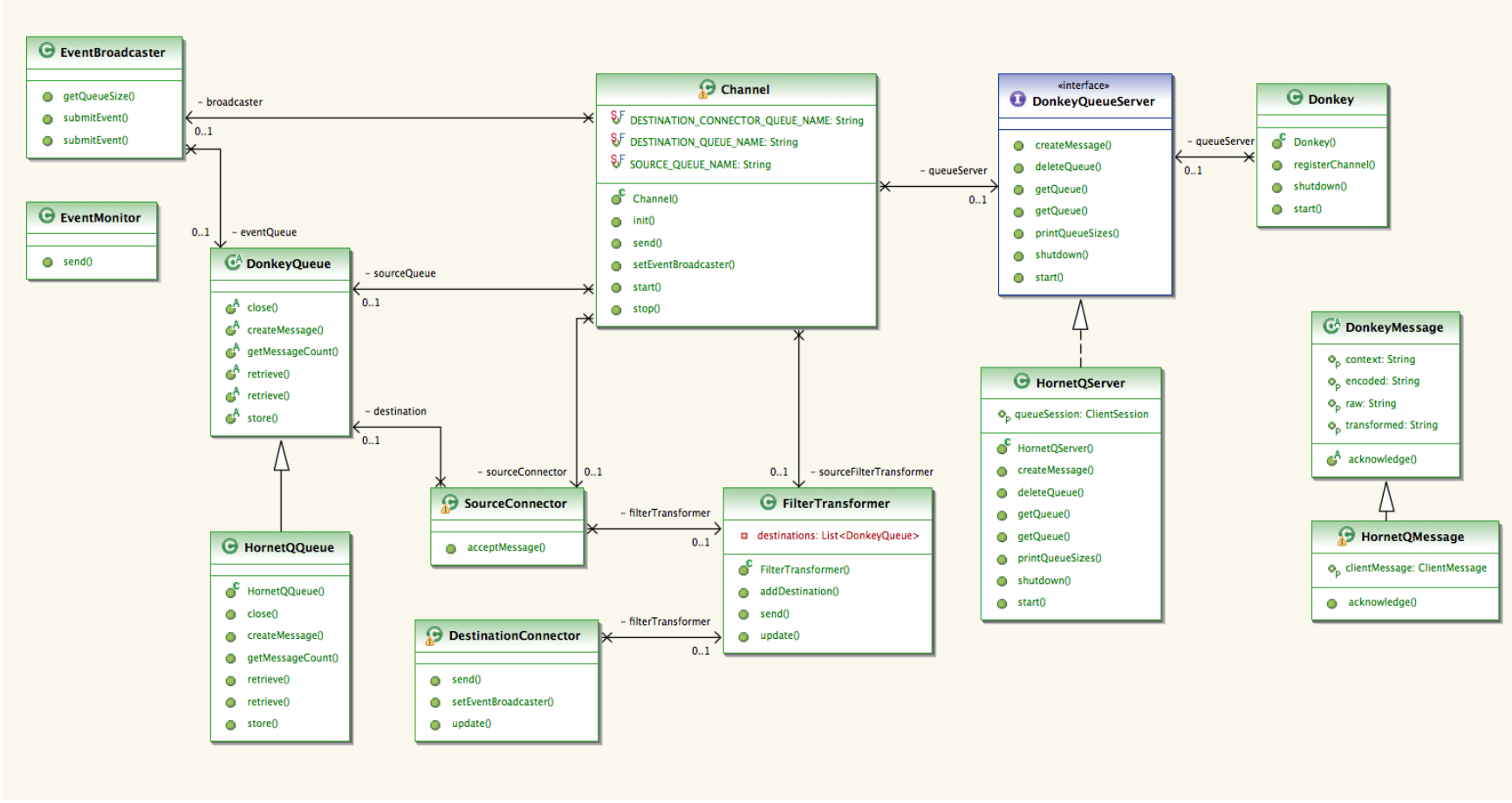
April 2011


Message Storage
Status: Received Filtered Sent Queued Errored
Store Contents: Yes No
Contents: Raw Transformed Encoded
Encrypt Data: Yes No
Message Pruning
Prune: Yes No
Status: Received Filtered Sent Queued Errored
Age (days): 7

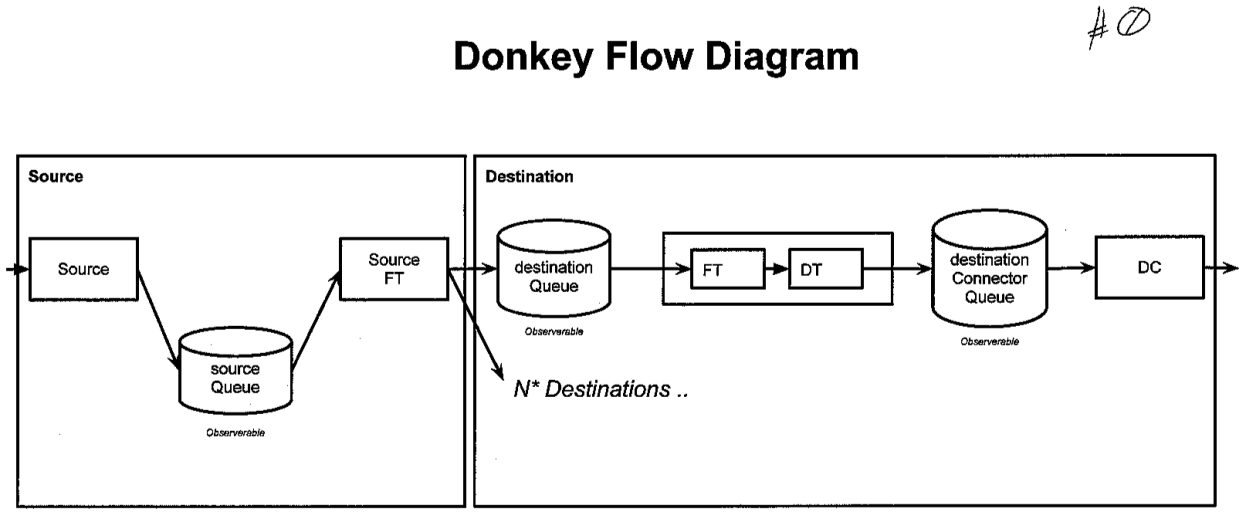
February 3, 2012



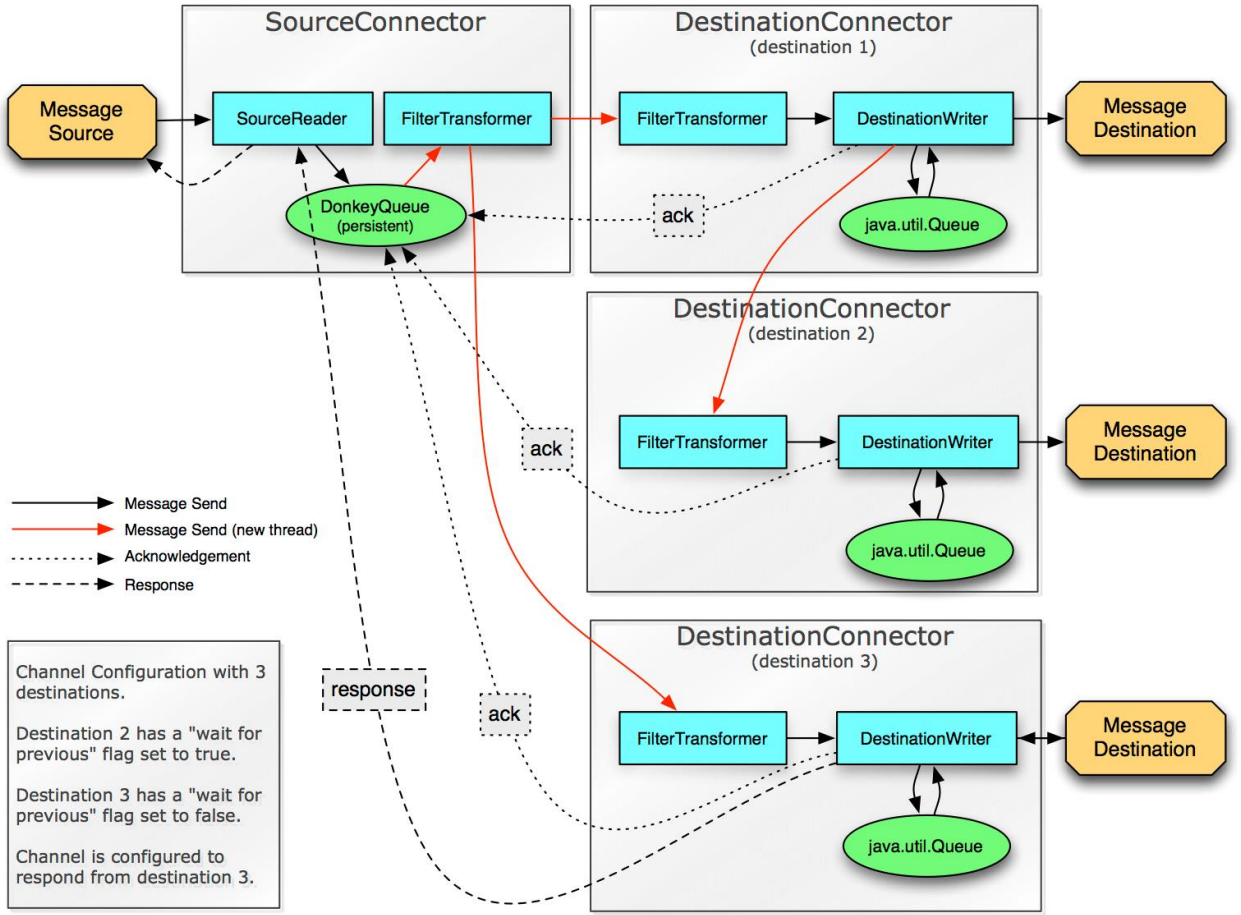
February 9, 2012



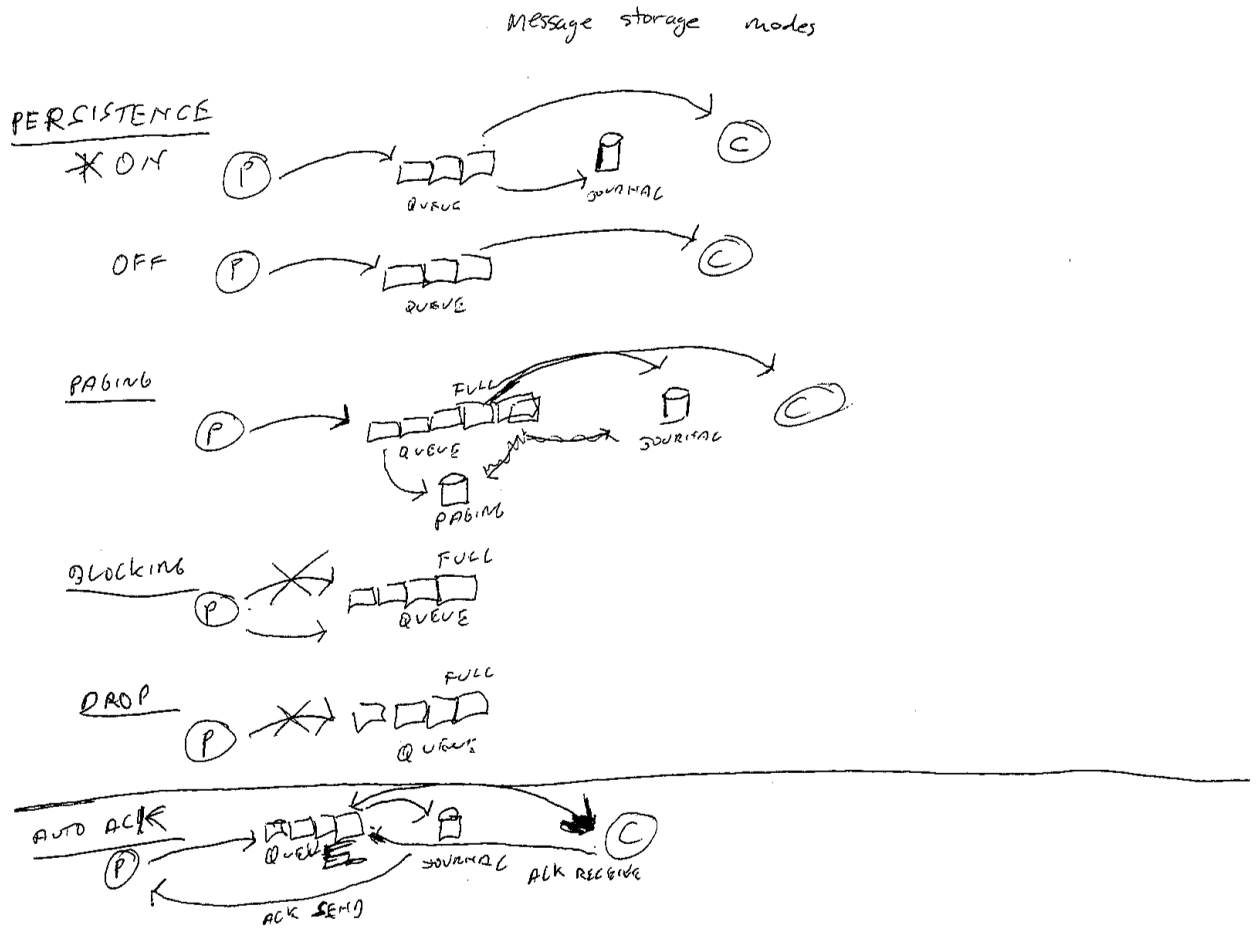
February 2012



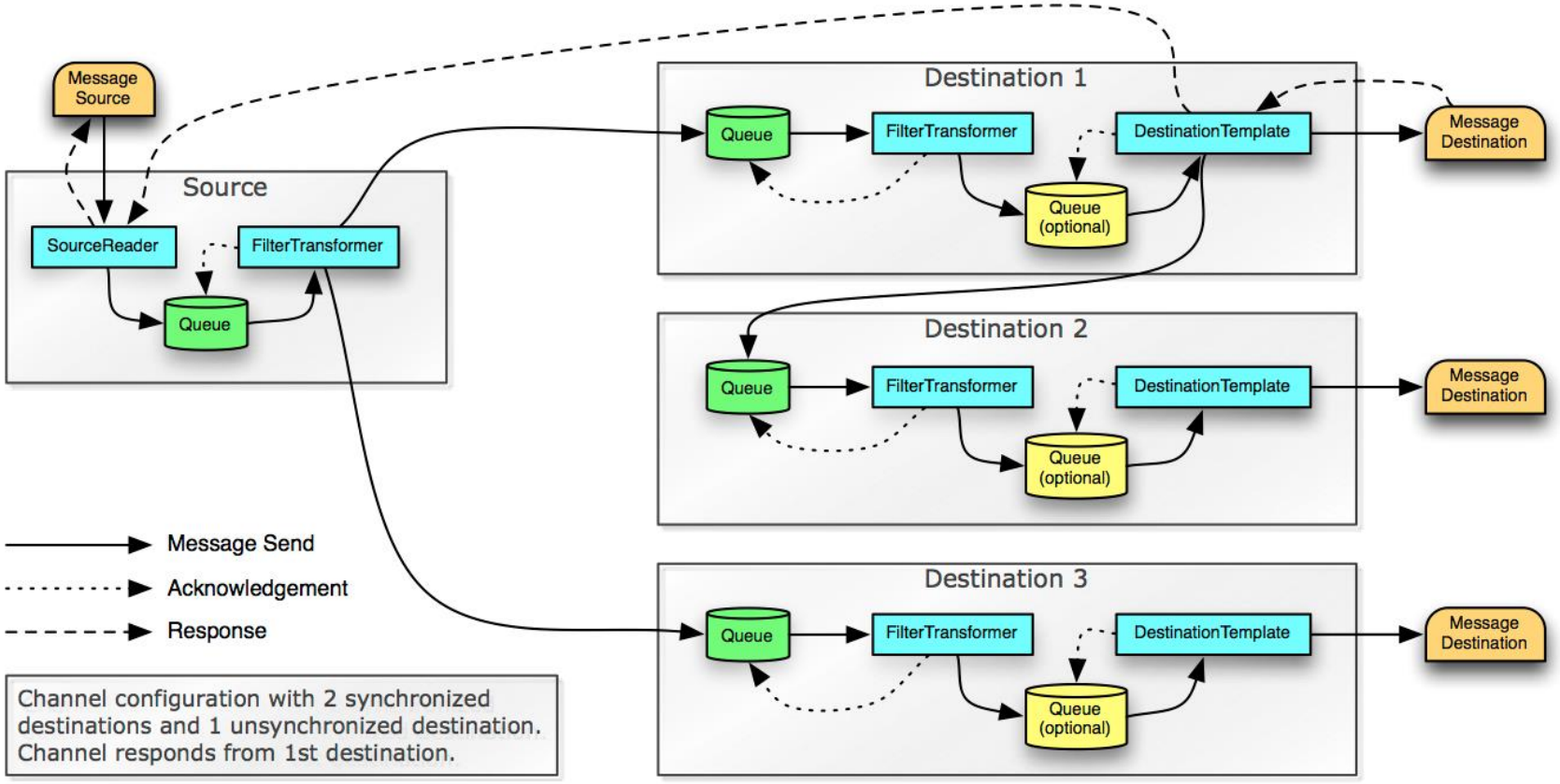
February 15, 2012



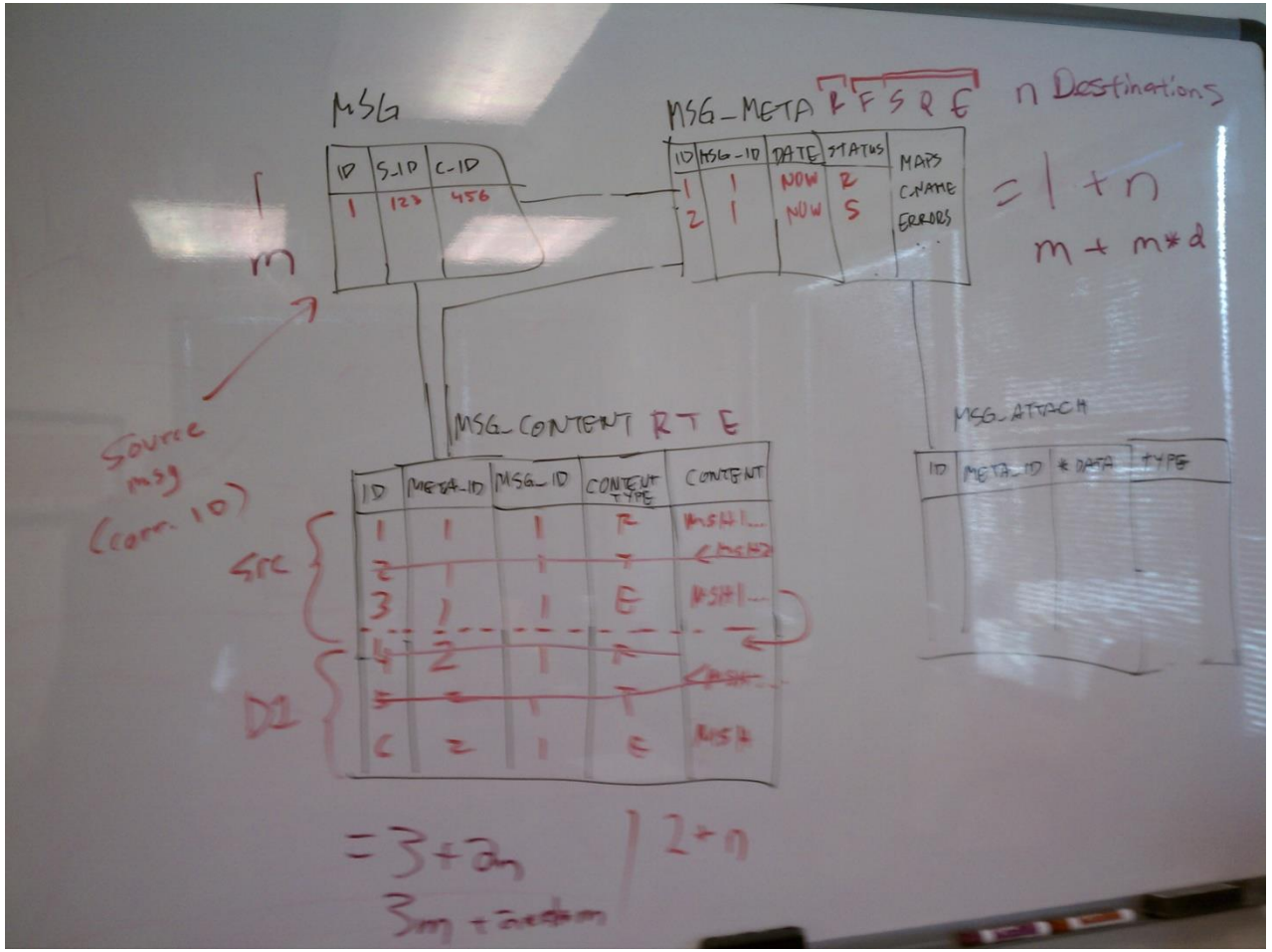
March 2012


Message storage modes
PERSISTENCE
ON
P
QUEUE
JOURNAL
C
OFF
P
QUEUE
C
PAGING
FULL
P
QUEUE
JOURNAL
C
PAGING
BLOCKING
FULL
P
QUEUE
DROP
FULL
P
QUEUE
AUTO ACK
P
QUEUE
JOURNAL
C
ACK RECEIVE
ACK SENT

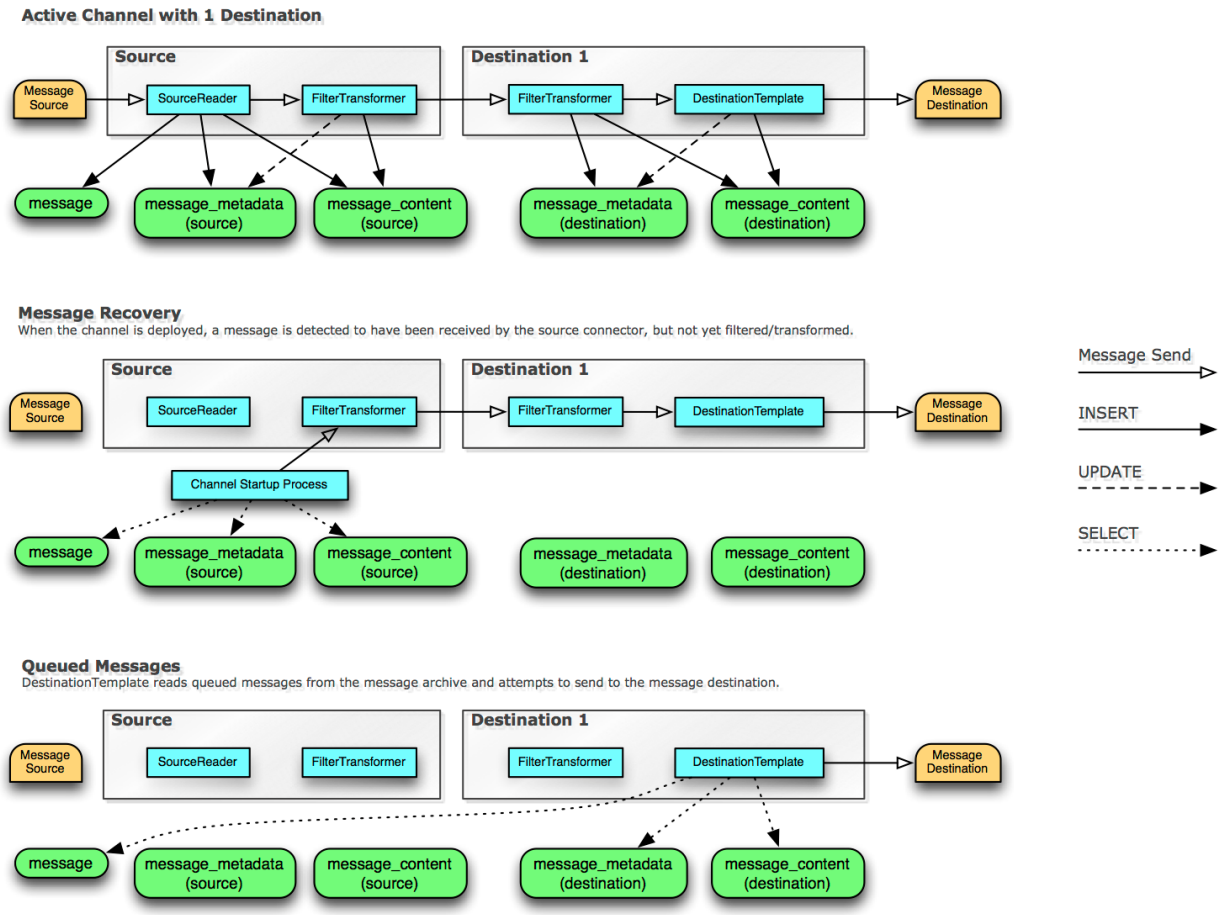
March 6, 2012


Message Source
Source
SourceReader
FilterTransformer
Queue
Destination 1
Queue
FilterTransformer
DestinationTemplate
Queue (optional)
Message Destination
Destination 2
Queue
FilterTransformer
DestinationTemplate
Queue (optional)
Message Destination
Destination 3
Queue
FilterTransformer
DestinationTemplate
Queue (optional)
Message Destination
Message Send
Acknowledgement
Response
Channel configuration with 2 synchronized destinations and 1 unsynchronized destination. Channel responds from 1st destination.

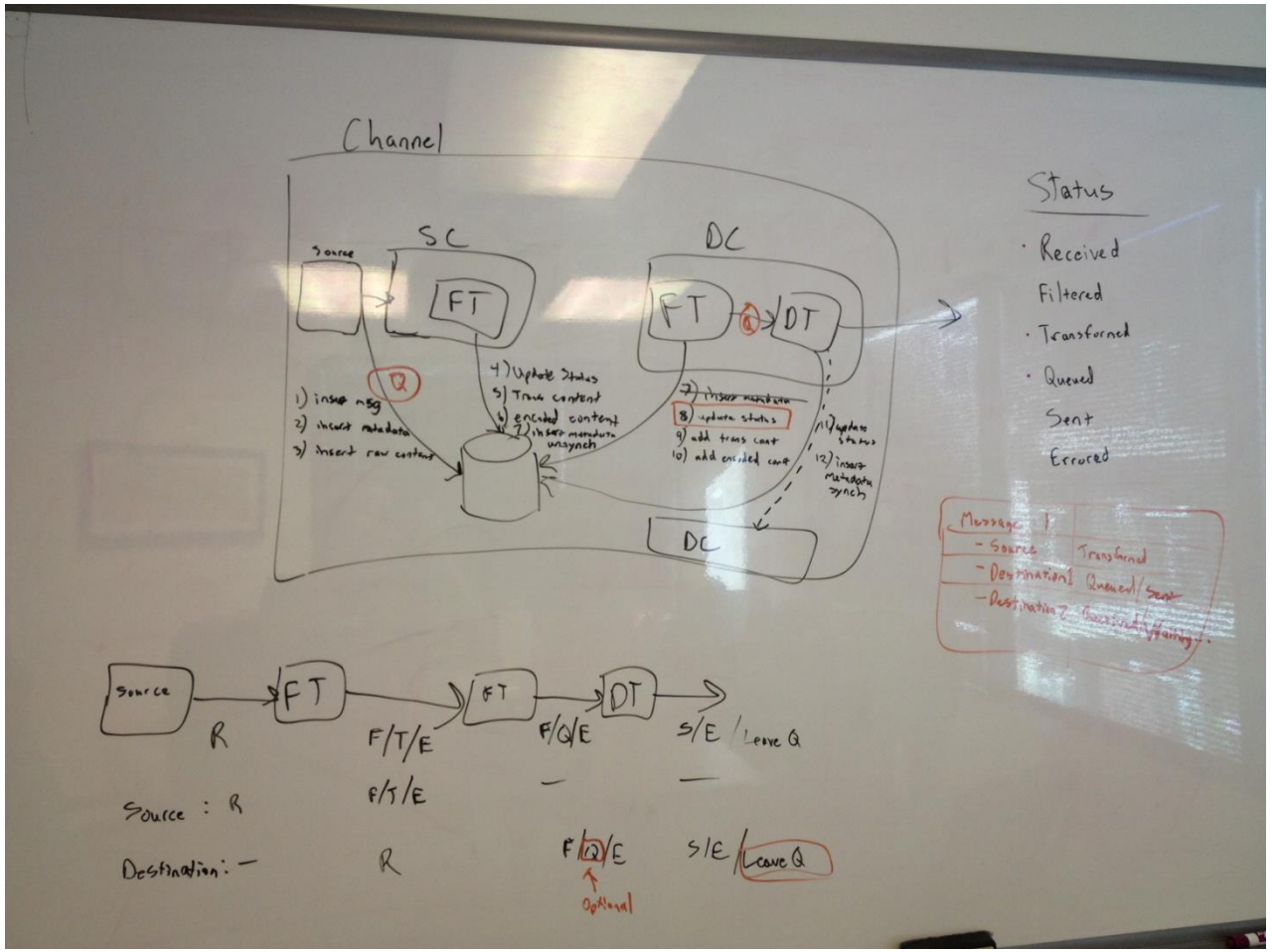
March 12, 2012



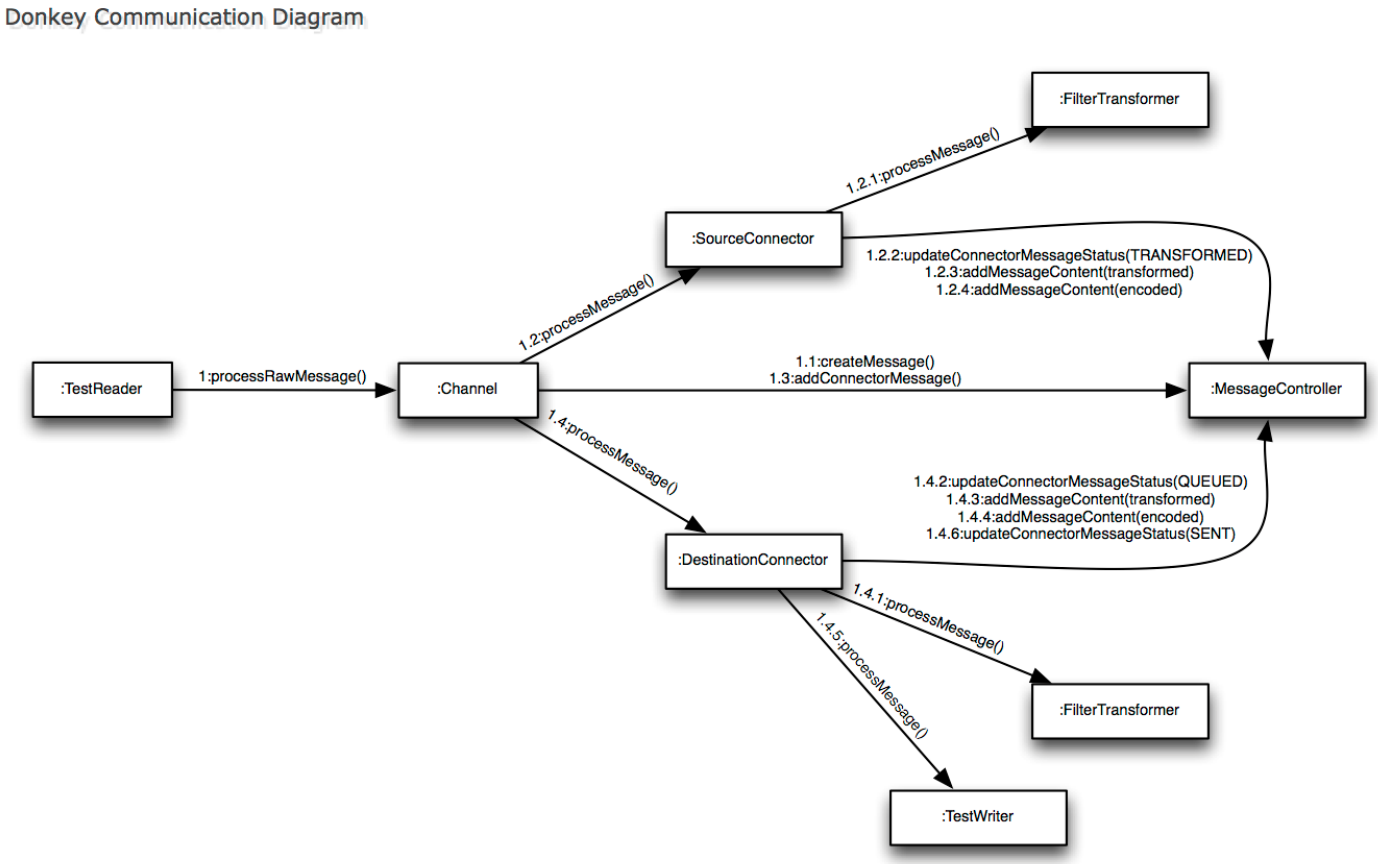
March 13, 2012



March 14, 2012



March 14, 2012



March 20, 2012