ECE444: Software Engineering

Requirements 4:Risk, Prototypes

Shurui Zhou

The Edward S. Rogers Sr. Department
of Electrical & Computer Engineering
UNIVERSITY OF TORONTO

Learning Goals (Last lecture)

- Understand tradeoffs of different documentation strategies
- Document requirements using use cases and user stories
- Evaluate the quality of a user story by INVEST

Product Requirement Document (PRD)

1. Goals
2. User Personas
3. User Stories
4. Functional Requirements
5. Non-Functional Requirements
6. User interaction and design
7. Questions
8. Out of Scope

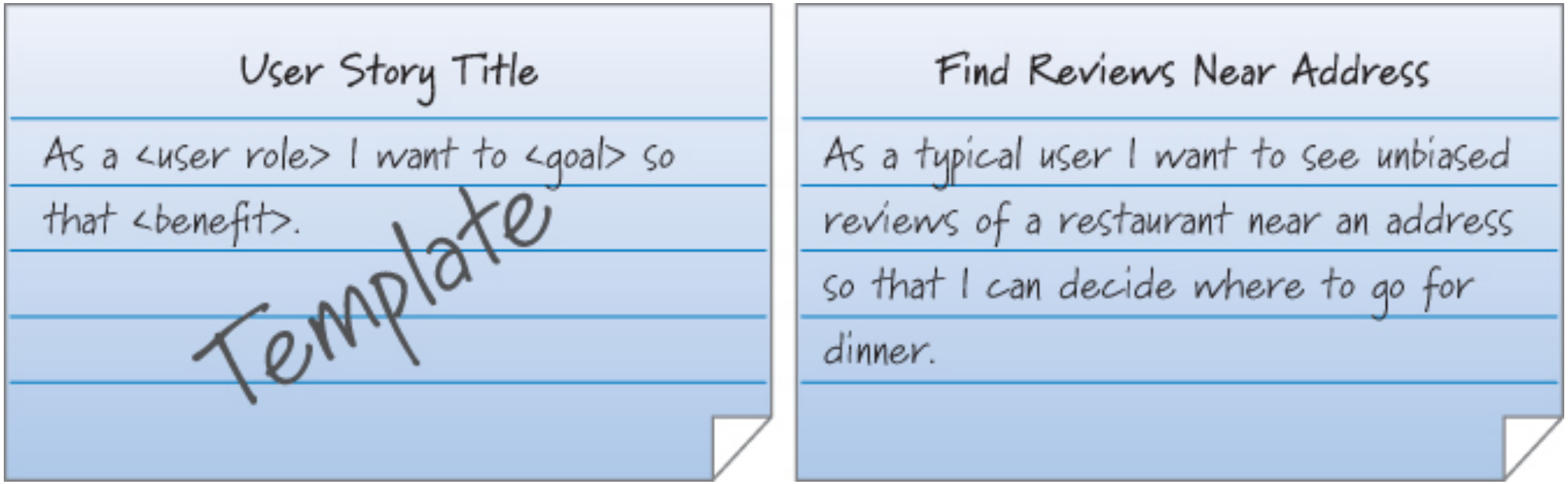
Learning Goals

- Understand how to organize user stories into story map
- Understand risk and its role in requirements, specifically how it can be identified, analyzed, and then mitigated/handled in system design.
- Low/high fidelity design

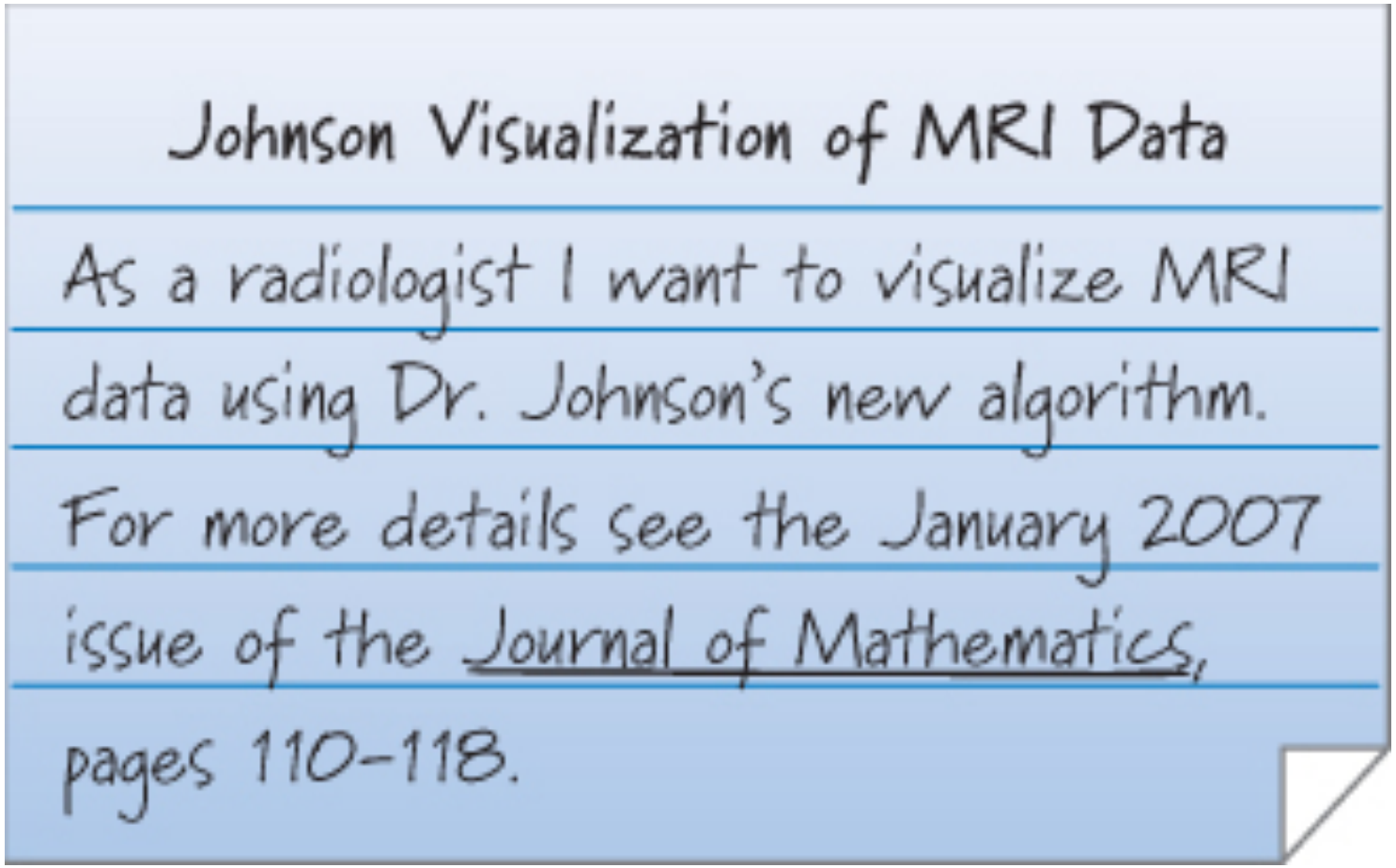
User Stories

card	a brief, simple requirement statement from the perspective of the user
conversation	a story is an invitation for a conversation
confirmation	each story should have acceptance criteria

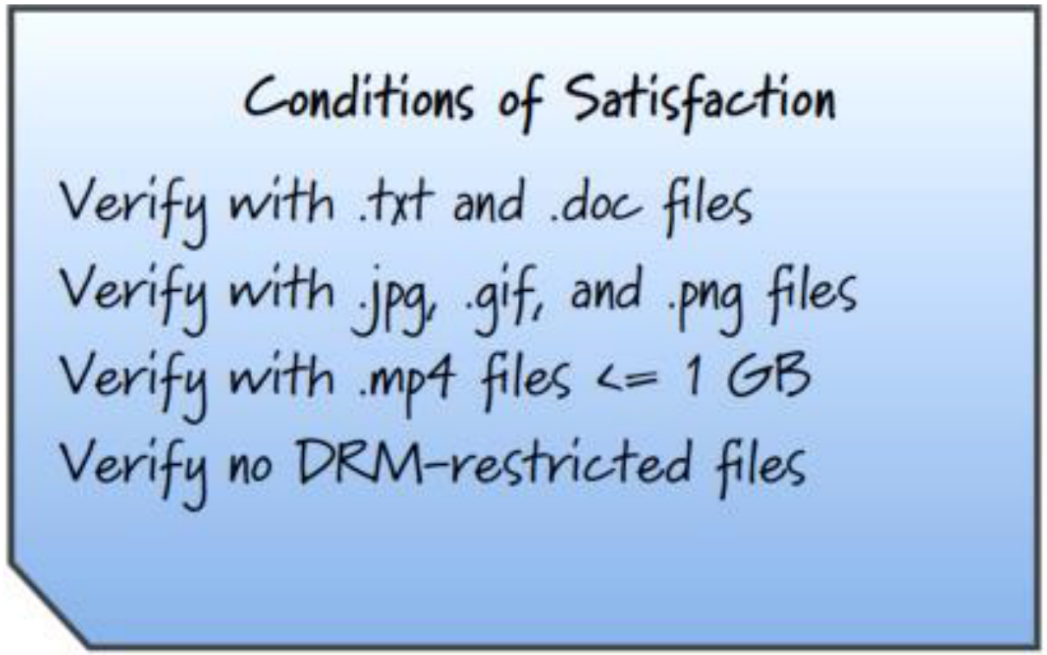
User Story Example - Card



User Story Example - Conversation



User Story Example - Confirmation



Non-Functional Requirements

- Security
- Performance
- Reliability
- Usability
- …

Some might be global, some local
– All responses should be below 3 seconds
– The wheel's revolutions per minute should be sampled 200 times per second to prevent aliasing effects

It is hard to reconcile global properties with agile principles

Non-Functional Requirements

Internationalization

As a user I want an interface in English, a Romance language, and a complex language so that there is high statistical likelihood that it will work in all 70 required languages.

Web Browser Support

System must support IE8, IE9, Firefox 6, Firefox 7, Safari 5, and Chrome 15.

Good Requirements

Traditional

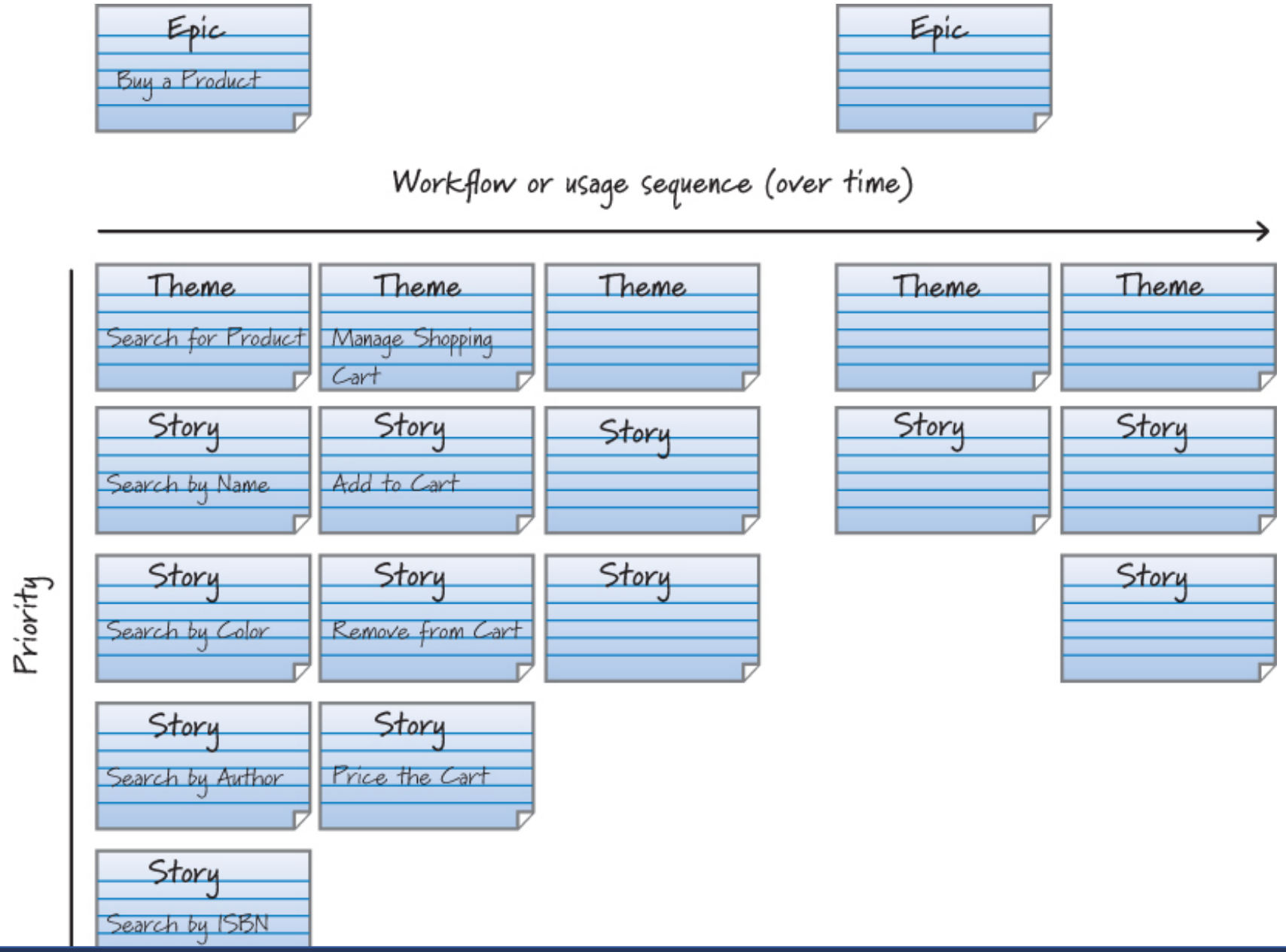
- MECE. The requirements form a **m**utually **e**xclusive and **c**ollectively **e**xhaustive expression of the user needs and wants
- Complete. Each requirement must fully describe the capability to be delivered
- Unambiguous. All readers of a requirement should arrive at a single, consistent interpretation of it
- Verifiable. It should be possible to objectively determine whether the system properly implements each requirement
- Consistent. A requirement must not conflict with other requirement

User stories (INVEST)

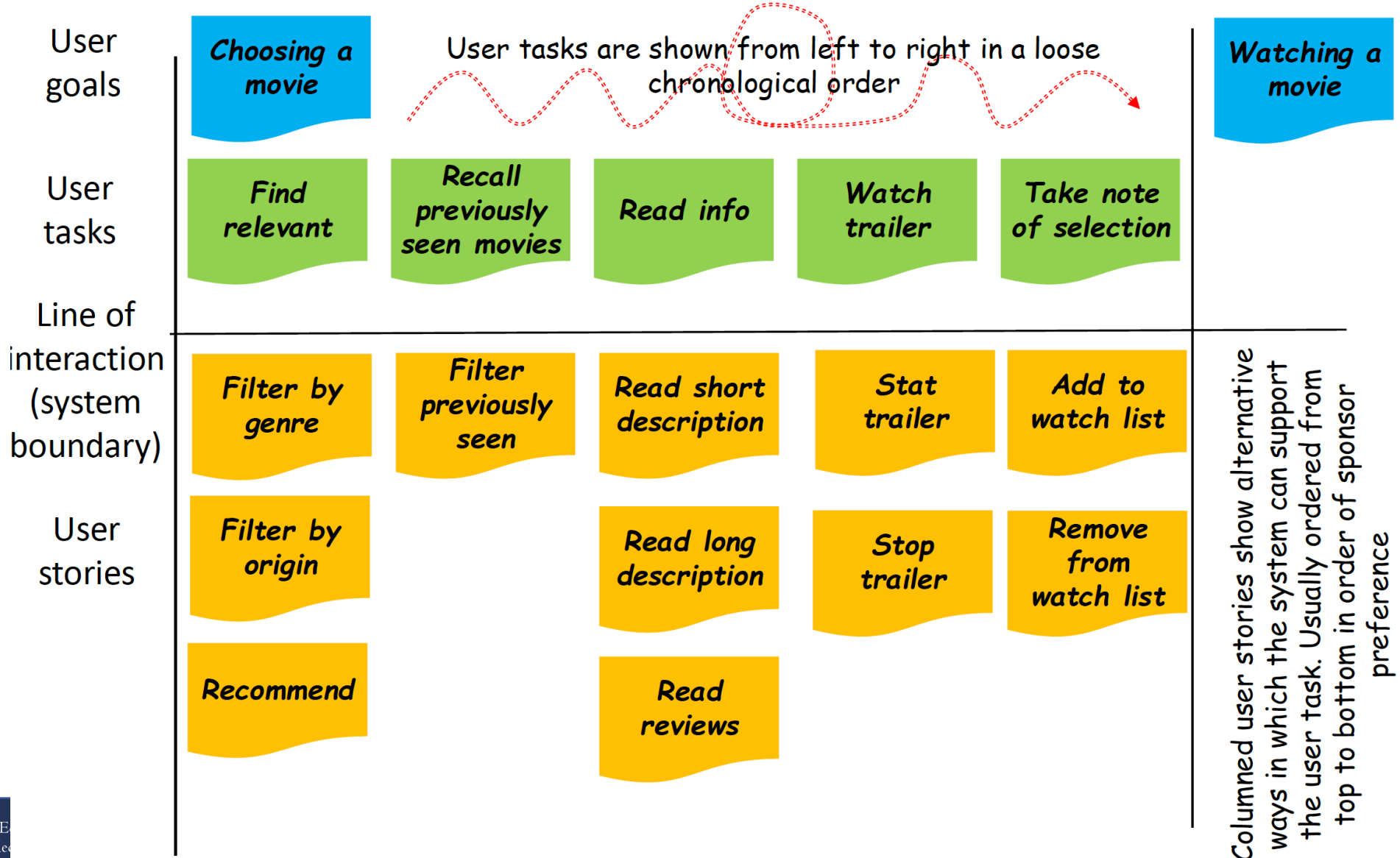
- **I**ndependent. The requirement can be developed and tested on its own
- ***N****egotiable (Refinable). The requirement is a promise to have a conversation in due time to define the details of whatever is being built. Is more about learning than negotiation*
- **V**aluable. The requirement must provide a benefit the customer could appreciate
- **E**stimable. It should be possible for the team to forecast the effort it will require to implement it
- **S**mall. The requirement should be small enough to be able to be completed in an iteration
- **T**estable. The requirement must provide enough information to make it clear how to verify it will be verified

Story Mapping

- Epic
  - Theme
    - Story



Story map for choosing a movie



From goals to story maps

1. Consolidate goals across scenarios (there could be more than one scenario for every given goal)
2. Create a story map column for each consolidated goal
3. Enumerate the tasks you find in the scenarios in the most likely chronological order, do not concern with an strict order, shortcuts, repetitions, etc. Eliminate duplicates
4. Are there missing tasks?
    1. Are there tasks that should precede or succeed any one of the ones you have already included listed?
    2. Are there important task variations that should be considered?
5. What support will the system provide to the user tasks above? List the user stories under the corresponding task in order of preference
    1. Are there alternative ways to support the task?
    2. Does the solution require that the user perform some additional task?
    3. Are there user stories that should precede or succeed the current one?

Risk Management

Risk


Tony Webster
@webster
Follow
I appreciate the honesty.
Pick a password
Don't reuse your bank password, we didn't spend a lot on security for this app.
At least 6 characters
your password
Continue
8:20 PM - 15 Sep 2018
5,868 Retweets 15,672 Likes
58
5.9K
16K

What are risks?

- A **risk** is an uncertain factor that may result in a loss of satisfaction of a corresponding objective

For example…

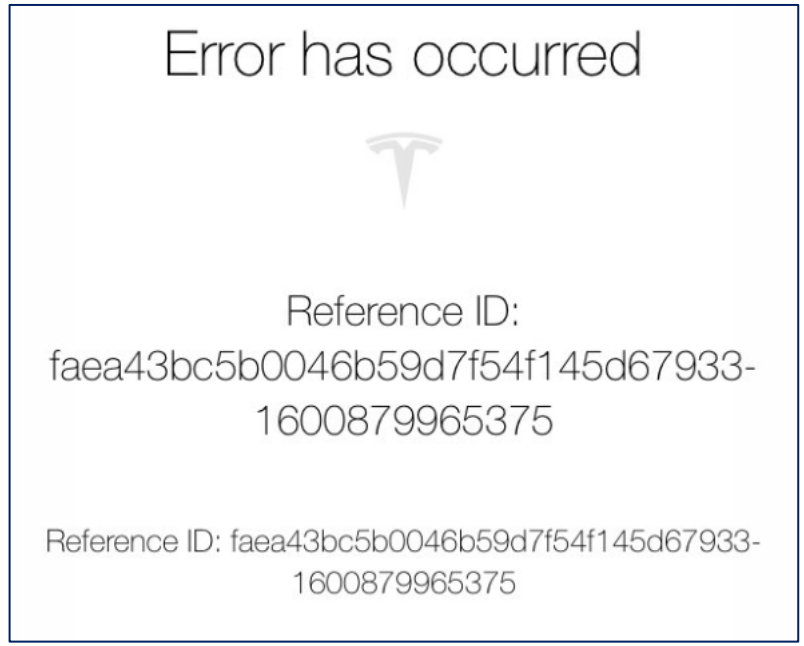
- System delivers a radiation overdose to patients (Therac-25, Theratron-780)
- Premier Election Solutions vote-dropping “glitch”

Kishore Gopalakrishna @KishoreBytes · Sep 23

I am stranded and cant get into my **Tesla**.

- App won't open
- Keycard is not working
- Emergency roadside assistance- one hour wait time. There is no call me back option.

This is the **error** message. This makes me feel better about the **error** messages in oss projects.



How to assess the level of risk?

- Risks consist of multiple parts:
  - Likelihood of failure
  - Negative consequences or impact of failure
  - Causal agent and weakness (in advanced models)
- Risk = Likelihood x Impact

Aviation failure impact categories

- **No effect** – failure has no impact on safety, aircraft operation, or crew workload
- **Minor** – failure is noticeable, causing passenger inconvenience or flight plan change
- **Major** – failure is significant, causing passenger discomfort and slight workload increase
- **Hazardous** – high workload, serious or fatal injuries
- **Catastrophic** – loss of critical function to safely fly and land

Risk assessment matrix

- MIL-STD-882E

TABLE III. Risk assessment matrix

RISK ASSESSMENT MATRIX				
SEVERITY / PROBABILITY	Catastrophic (1)	Critical (2)	Marginal (3)	Negligible (4)
Frequent (A)	High	High	Serious	Medium
Probable (B)	High	High	Serious	Medium
Occasional (C)	High	Serious	Medium	Low
Remote (D)	Serious	Medium	Medium	Low
Improbable (E)	Medium	Medium	Medium	Low
Eliminated (F)	Eliminated			

https://myclass.dau.edu/bbcswebdav/institution/Courses/Deployed/TST/TST303/Student_Materials/Student%20Lessons%20%28PDF%29/L12S-RIO/Lesson%20Material/MIL-STD%20882E%20%28Extract%29

Risk Mgmt: Waterfall vs Agile

- Longer development and planning cycle
- In waterfall projects: Testing at the end of the project
- Less responsive to changes
- Prescribed comprehensive processes to identify, assess and review risks and assign risk responses

- Short development cycles and quick delivery
- Testing is part of the development cycle
- Business people are often part of the team which reduces risks
- High responsiveness to changes
- Most frameworks do not prescribe risk management processes and techniques which requires the project team to select and adapt adequate measures

Risk Response Strategies

- Accept the risk – for low likelihood or low impact risks, or where cost of mitigation precludes system
- Transfer the risk – push the risk outside the system boundary
- Mitigate the risk – introduce active countermeasures
  - Reduce likelihood of failure
  - Reduce severity of failure
- Avoid the risk – redesign so that risk cannot occur

Risk analysis example (Time Keeper)

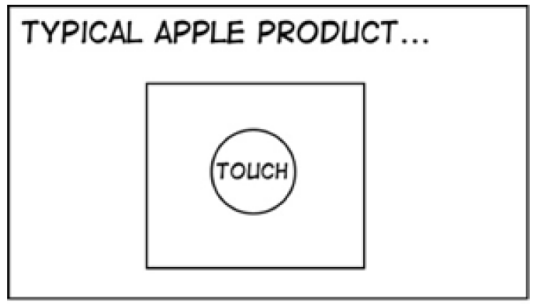
	Risk	Probability	Impact	Solution
1	Application crashes	Low	Medium	Introduce Long-term stability test
2	Inappropriate auto-scheduling	Medium	Low	Adjust auto-generated schedule manually
3	Outdated integration	Low	Low	Ignore

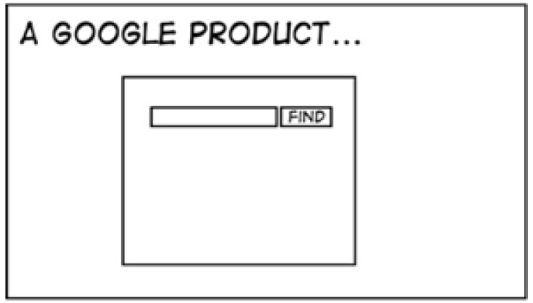
Prototypes, Mockups, Stories

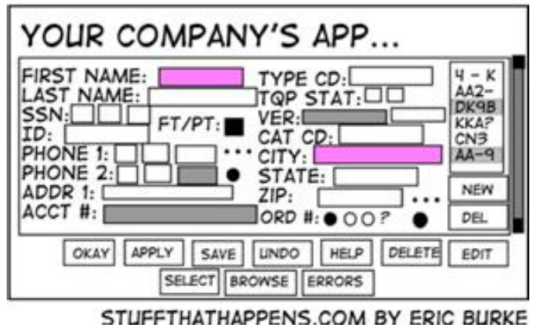
Product Requirement Document (PRD)

1. Goals
2. User Personas
3. User Stories
4. Functional Requirements
5. Non-Functional Requirements
6. User interaction and design
7. Questions
8. Out of Scope

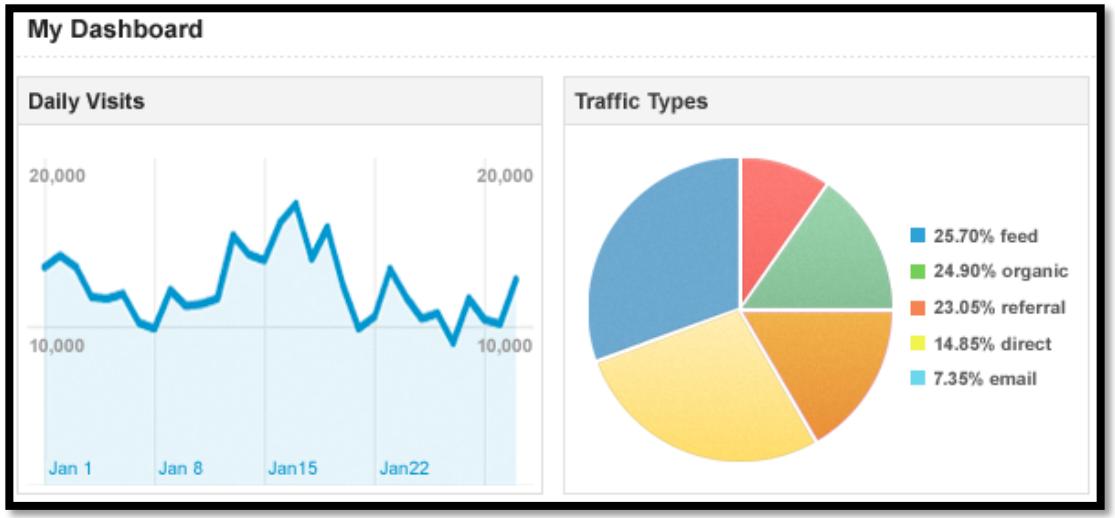
How should the product look?

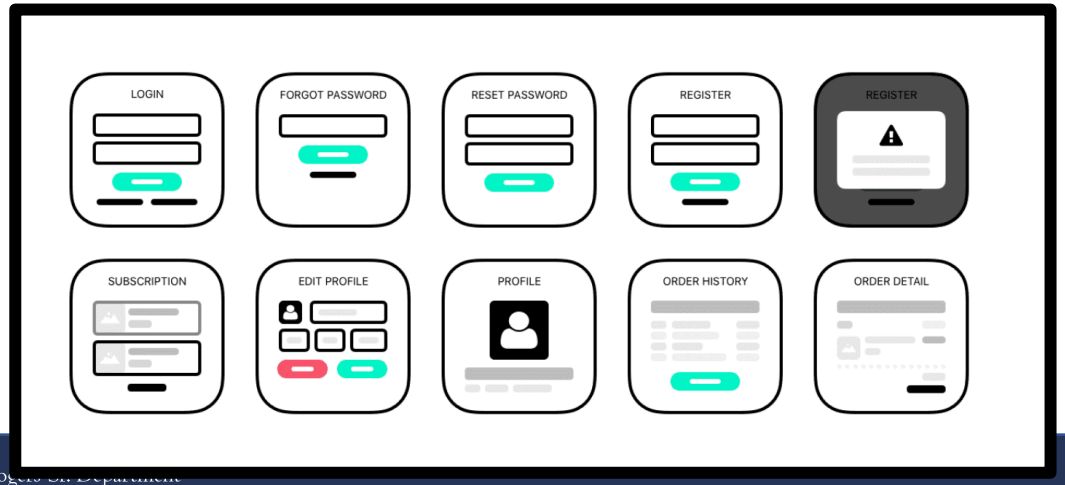






High- vs low- fidelity mockups



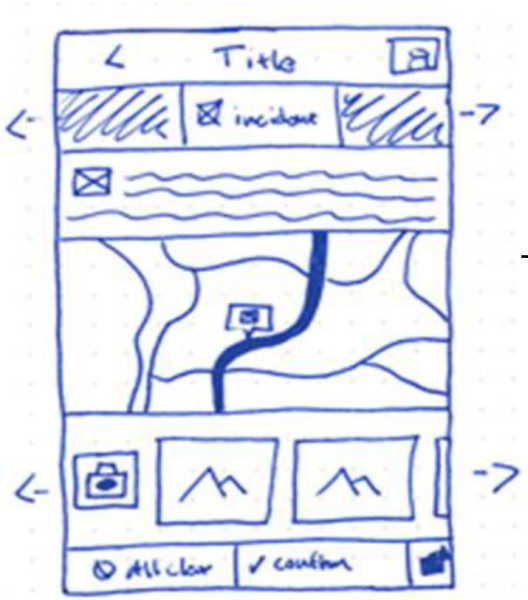


Time on Site by Country

Country/Territory	Visits	Avg. Time on Site
United States	67,445	00:01:54
United Kingdom	18,948	00:01:37
India	8,882	00:00:58
Canada	6,371	00:01:02
Germany	5,845	00:00:32
France	5,243	00:0038

Wireframes, low, and high fidelity prototypes

Wireframe



Real state allocation

Low fidelity prototype

Colors and fonts

High fidelity prototype

Navigation & mock results

Wireframe, Prototype, Mockup

	Fidelity	Cost	Use	General traits
Wireframe	low fidelity	$	Documentation, quick communication	Sketchy, black, white & grey representation of the interface
Prototype	**middle** to high fidelity	$$$	User testing, reusable backbone of the interface	Interactive
Mockup	middle to **high** fidelity	$$	Gathering feedback and getting buy-in from stakeholders	Static visualization

https://designmodo.com/wireframing-prototyping-mockuping/

	HIGH-FIDELITY PROTOTYPE	LOW-FIDELITY PROTOTYPE
Interactivity		
Clickable links and menus	Yes: Many or all are clickable.	No: Targets do not work.
Automatic response to user's actions	Yes: Links in the prototype are made to work via a prototyping tool (e.g., InVision, PowerPoint).	No: Screens are presented to the user in real time by a person playing "the computer."
Visuals		
Realistic visual hierarchy, priority of screen elements, and screen size	Yes: Graphics, spacing, and layout look like a live system would look (even if the prototype is presented on paper).	No: Only some or none of the visual attributes of the final live system are captured (e.g., a black-and-white sketch or wireframe, schematic representation of images and graphics, single sheet of paper for several screenfuls of information). Spacing and element prioritization may or may not be preserved.
Content and Navigation Hierarchy		
Content	Yes: The prototype includes all the content that would appear in the final design (e.g., full articles, product-description text and images).	No: The prototype includes only a summary of the content or a stand-in for product images.

Mockups, Prototypes, Stories

- Humans: better at recognizing whether a solution is correct than solving the problem from a blank page.
- Mock-ups/prototypes help explore uncertainty in the requirements.
  - Validate that we have the right requirements.
  - Elicit requirements at the “borders” of the system.
  - Assert feasibility of solution space.
  - Get feedback on a candidate solution.
- “I’ll know it when I see it”

Rapid prototyping

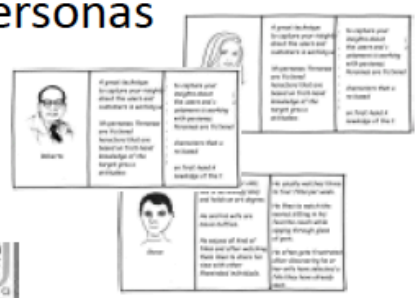
- Throw-away: developed to learn more about a problem, not intended for actual use.
- Evolutionary: intended to be incorporated into the final product.

https://images.app.goo.gl/D54VKKtS4Bpgob3W8

Summary

Contextual design, analytics, customer journeys

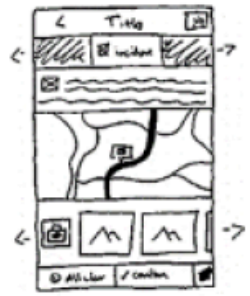
Personas



Scenarios

Persona data	Routine tasks
Design implication	Make user's most common tasks clear and apparent in the most visible area of the screen
Persona data	Infrequent user
Design implication	Provide wizards or defaults for the most common tasks
Persona data	Frequent interruptions
Design implications	Implement mechanisms to pause and recover from where the task was left off
Persona data	Bright noisy environment
Design implications	Make warnings clearly visible, use high contrast. Do not use audible feedback
Persona data	Low computer skills

Design tactics



Wireframes & Prototypes

		Roles			
		User	Admin		Parent
Personas	Peter	x			x
	Roberto				
	Mary	x			
Scenarios	S1	x			
	S2	x			
	Sk				x
	Sn		x		

Story map

As a user I want see a list of movies of a given genre

User story

Requirement analysis

System Design

Summary

- User stories and story mapping
- Risk analysis
- Using prototypes to enhance discussions and decision making