Requirements prioritization, negotiation and decision making

SEng 321

Outline

- Why prioritize requirements?
- Difficulties in prioritizing requirements
- Methods for requirements prioritization

Difficult task

- Different stakeholders may have different priorities
- Organizations lack systematic data, metrics or techniques to help the prioritization process
- Often carried out informally
- Research shows that few companies know how to establish and communicate priorities

However important

- Prioritizing requirements helps:
  - Making acceptable tradeoffs among goals of quality, cost and time-to-market
  - Project planning in allocating resources based on requirements importance to the project as a whole

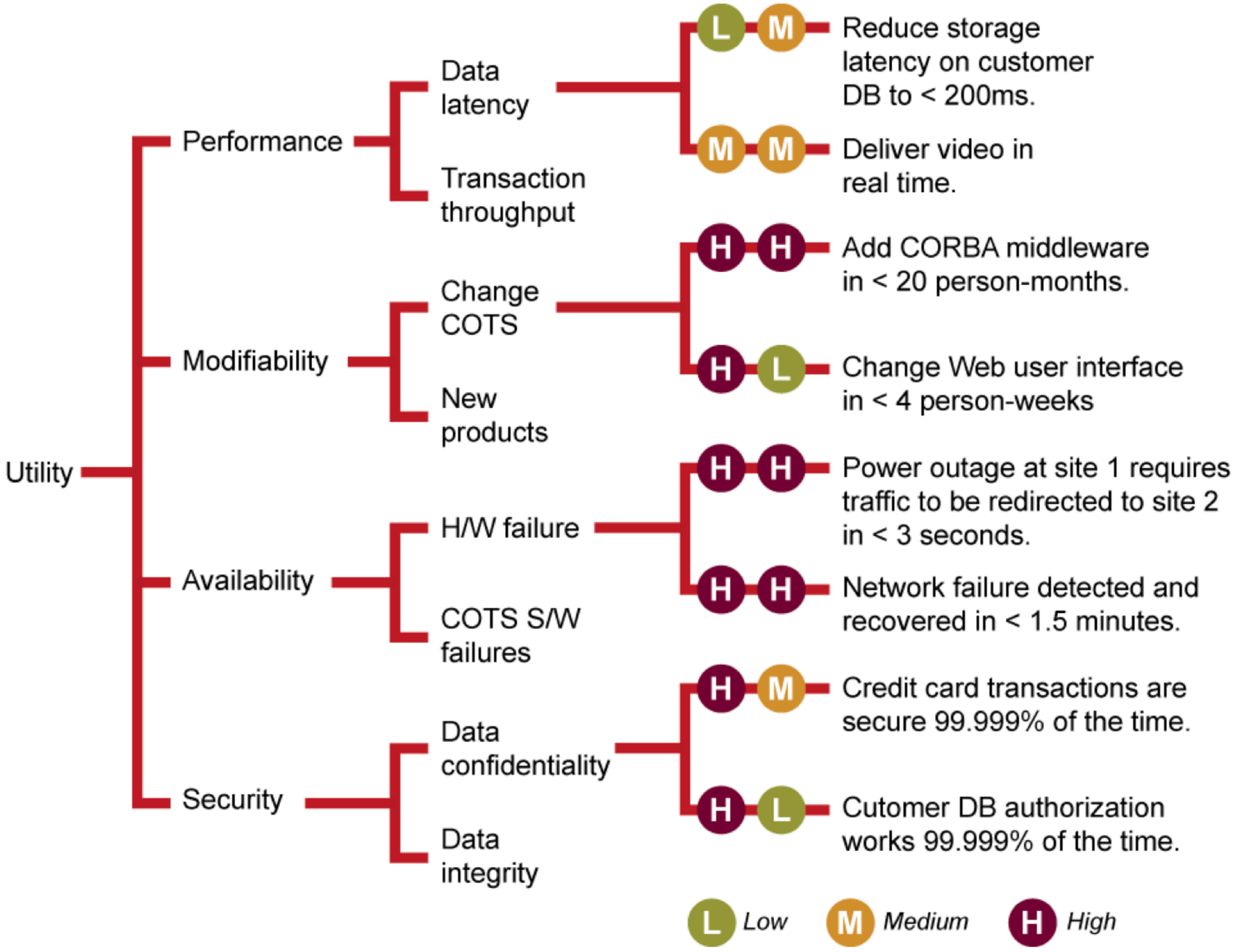
Six issues:

- The needs of the customers
- The relative importance of requirements to the customers
- The timing at which capabilities need to be delivered
- Requirements that serve as predecessors for other requirements and other relationships among requirements
- Which requirements must be implemented as a group
- The cost to satisfy each requirement

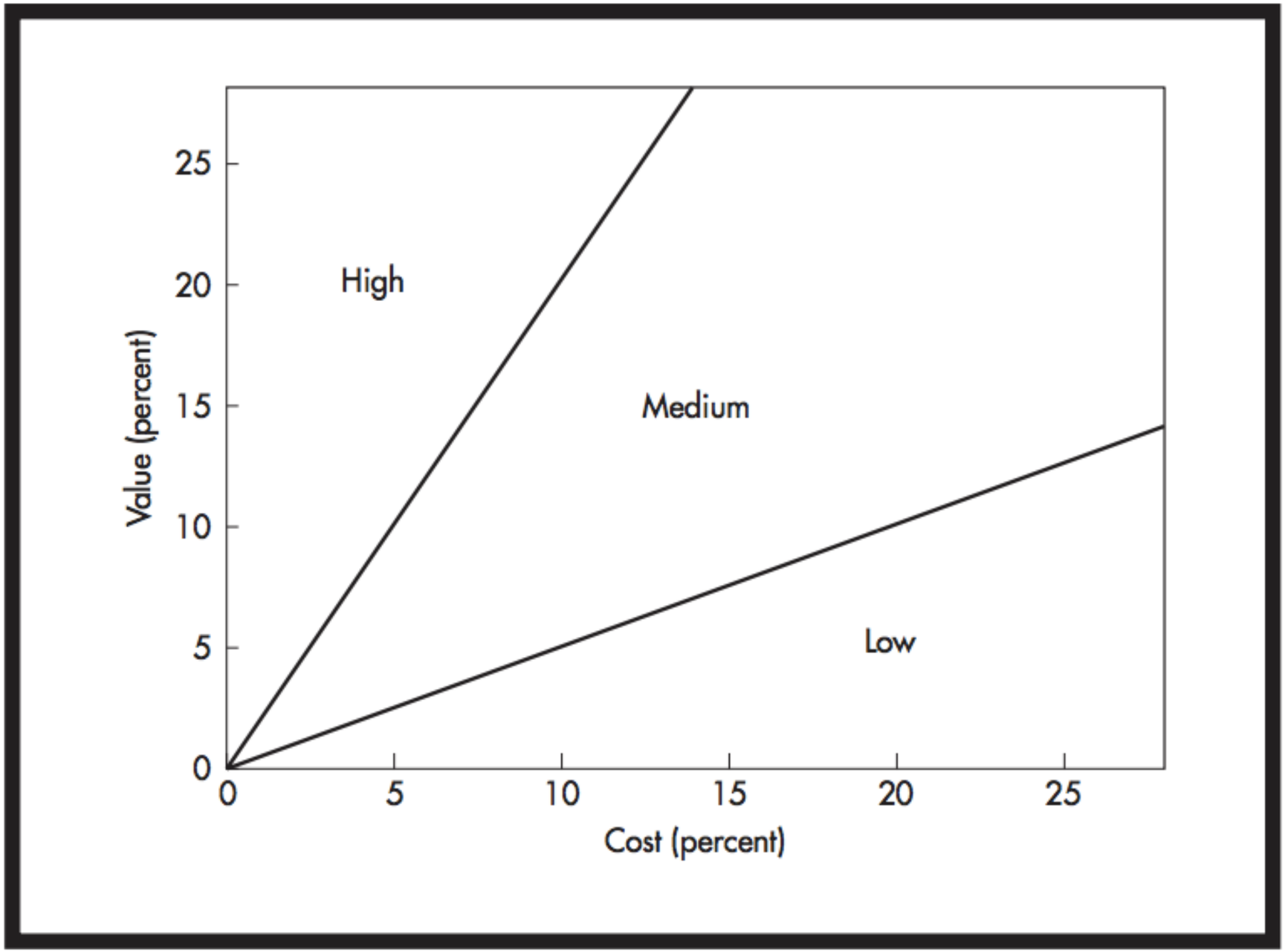
A process of prioritizing requirements

- Must be simple and fast, for industry adoption
- Yield accurate and trustworthy results
- Should consider issues of:
  - Importance of requirement to the user (maximize)
  - Cost of implementation (minimize)
  - Time-to-delivery (minimize)

Quality Attribute Prioritization


Utility
Performance
Modifiability
Availability
Security
Data latency
Transaction throughput
Change COTS
New products
H/W failure
COTS S/W failures
Data confidentiality
Data integrity
L M
Reduce storage latency on customer DB to < 200ms.
M M
Deliver video in real time.
H H
Add CORBA middleware in < 20 person-months.
H L
Change Web user interface in < 4 person-weeks
H H
Power outage at site 1 requires traffic to be redirected to site 2 in < 3 seconds.
H H
Network failure detected and recovered in < 1.5 minutes.
H M
Credit card transactions are secure 99.999% of the time.
H L
Cutomer DB authorization works 99.999% of the time.
L Low
M Medium
H High

Using the cost-value tradeoffs to make decisions



A quantitative, cost-value approach

Calculate return on investment by:

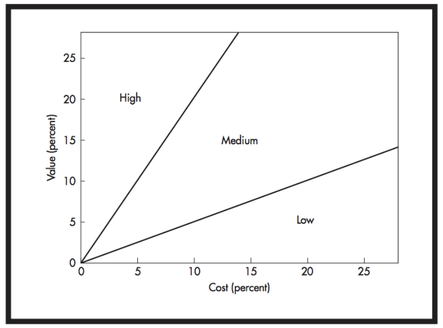
- assessing the value of each requirement
- assessing the cost of each requirement
- calculate the cost-value trade-off

Detailed practical steps

1. Requirements engineers check requirements for ambiguities, completeness, etc.

2. **Customers** estimate the relative **value** (**importance**) of candidate requirements

3. Experienced **software engineers** estimate the **cost** of candidate requirements
4. Plot these values on a **cost-value diagram**
5. Stakeholders use this map to analyze and make trade-offs



A cost-value approach

Calculate return on investment by:

- assessing the value of each requirement
- assessing the cost of each requirement
- calculate the cost-value trade-off

Difficulties:

- Hard to calculate absolute value/cost, relative values easier to obtain
- Interdependent requirements difficult to treat individually
- Inconsistencies or conflicts in priorities assigned by individual stakeholders

Could use:
The Analytical Hierarchy Process (AHP)

Step (to prioritize n requirements):

1. Set up the n requirements in the rows and columns of a nxn matrix
2. Perform pairwise comparisons of all the requirements according to the criterion
3. Use averaging over normalized columns to estimate the eigenvalues of the matrix:
   - Calculate the sum of the n columns in the comparison matrix
   - Divide each element in the matrix by the sum of the column the element is a member of, and calculate sums for each rows
   - Normalize the sum of the rows (divide each row sum with the number of requirements)
   - The result == priority information for each requirement

Pairwise comparison of requirements

Use a 1-9 scale with:

- $a_{ij} = 1$ if the two reqts are equal in importance
- $a_{ij} = 3$ if R_i is weakly more important than R_j
- $a_{ij} = 5$ if R_i is strongly more important than R_j
- $a_{ij} = 7$ if R_i is very strongly more important than R_j
- $a_{ij} = 9$ if R_i is absolutely more important than R_j

Example techniques that leverage AHP in RE

- As developed by Karlsson and Ryan
- use the Analytic Hierarchy Process to assess relative values and costs of requirements
- use cost-value diagrams to analyze and discuss candidate requirements
- Useful to managers for requirements triage and release planning

- See https://en.wikipedia.org/wiki/Analytic_hierarchy_process_%E2%80%93_car_example or https://bpmsg.com/ahp/ahp-calc.php?n=3&t=AHP+priorities&c[0]=Usability&c[1]=Security&c[2]=Speed

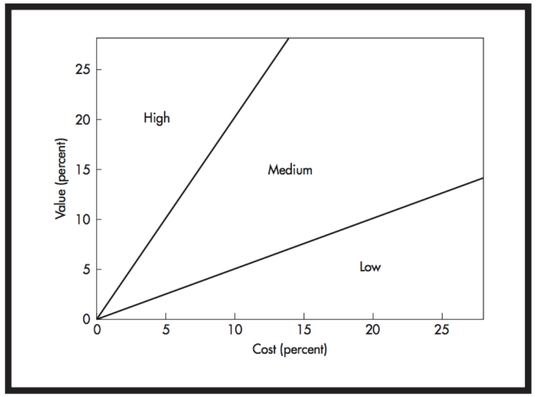
Detailed practical steps

1. Requirements engineers check requirements for ambiguities, completeness, etc.

2. **Customers** use *AHP's pairwise comparison* to estimate the relative **value** (**importance**) of candidate requirements

3. Experienced **software engineers** use *AHP's pairwise comparison* to estimate the **cost** of candidate requirements

5. Plot these values on a **cost-value diagram**
6. Stakeholders use this map to analyze and make trade-offs



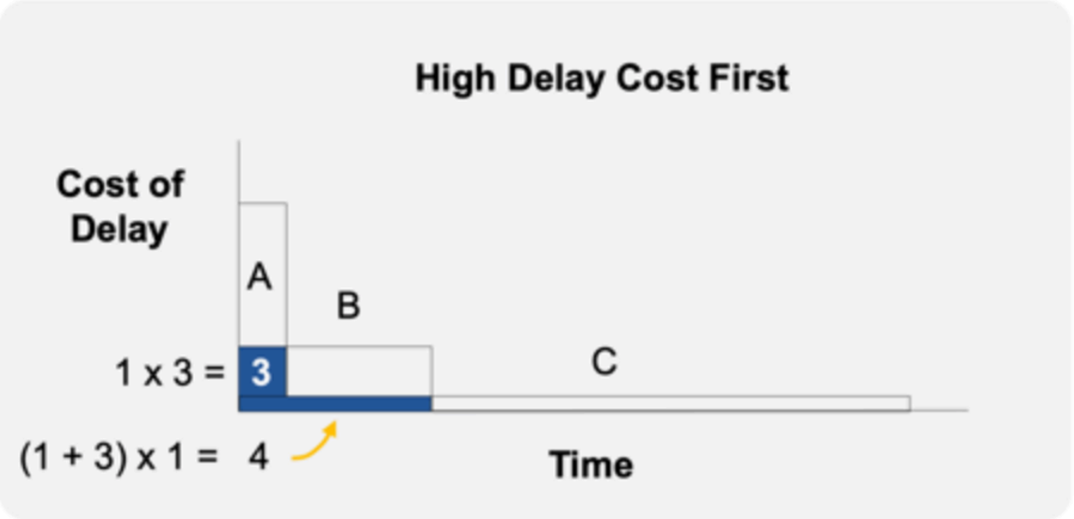
Applicability of method

- This cost-value technique has been applied successfully to industrial projects
- The book expands on this with the notion of “risk”, how hard it is to complete (“Quality Function Deployment”)
- Has some limitations in projects with:
  - large number of requirements, pairwise comparison can be tedious
  - many interdependencies between requirements
  - Distributed stakeholders

Prioritizing in Agile RE

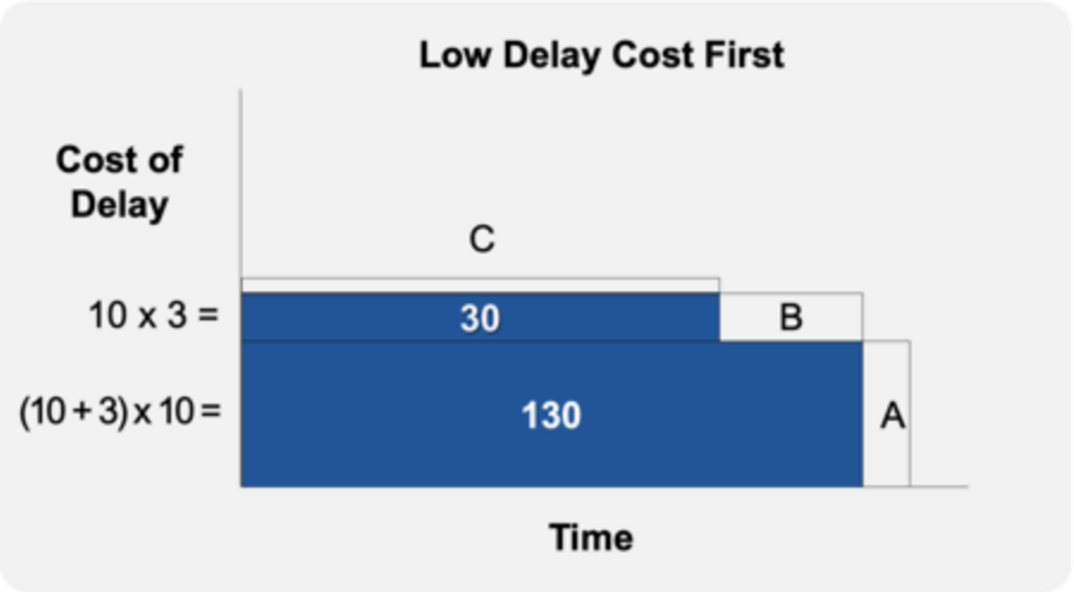
- Last Responsible Moment & Cost of Delay
  - *If we miss Dec 25, our Christmas feature is worthless.*
- WSJF - Weighted shortest job first - Reinertsen & SAFE
- Eisenhower matrix
- MoSCoW/Kano

WSJF



$$\text{WSJF} = \frac{\text{Cost of Delay}}{\text{Job Duration (Job size)}}$$

If effort and CoDs are different, do the Weighted Shortest Job First!



Feature	Cost of Delay	Duration	WSJF
A	10	1	10
B	3	3	1
C	1	10	0.1

Delay Cost

From *The Principles of Product Development Flow*, by Donald G. Reinertsen, Celeritas Publishing, © 2009 Donald G. Reinertsen

Figure 1. Applying the WSJF algorithm delivers the best overall economics

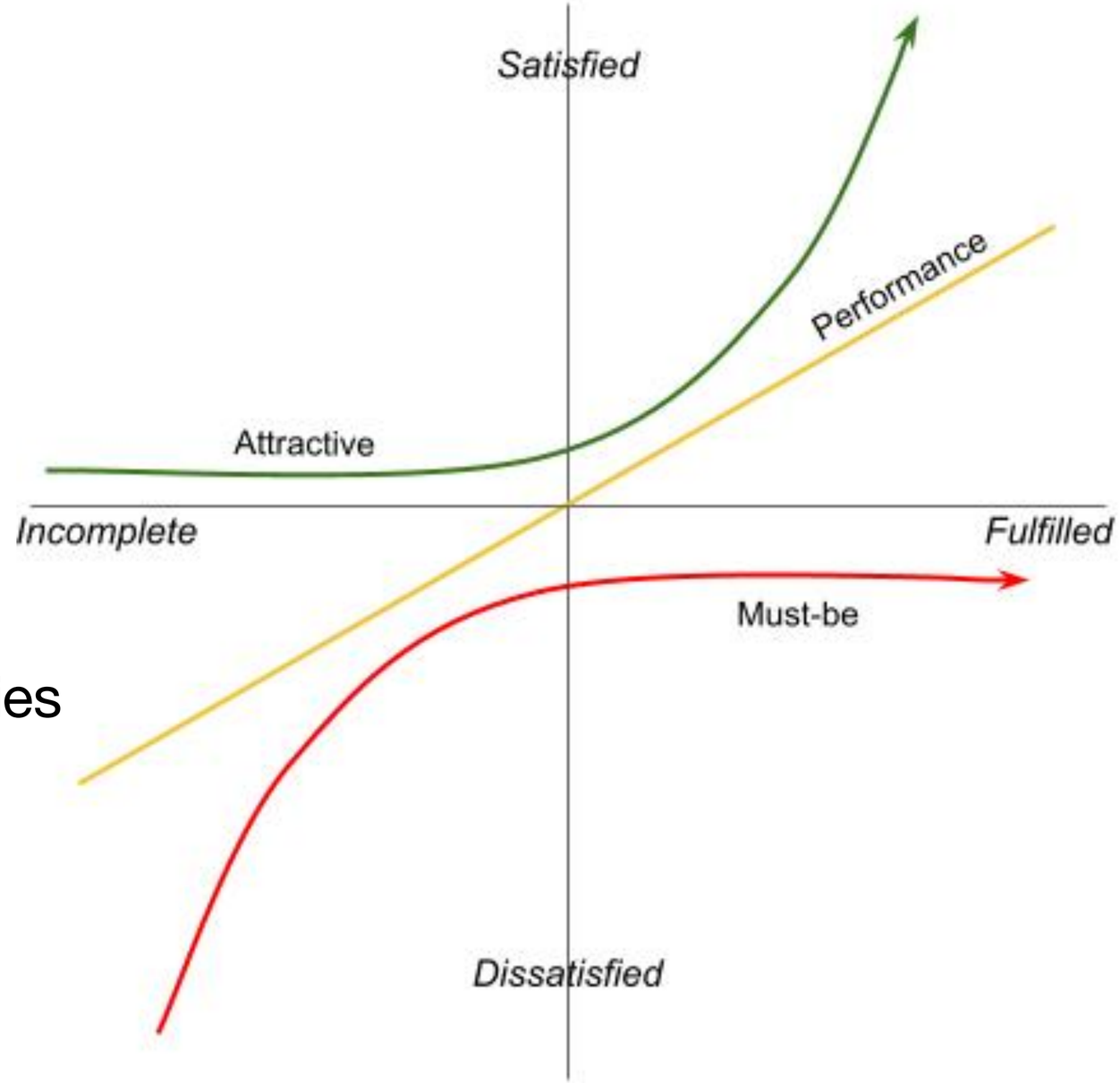
The Eisenhower Decision Matrix

MoSCoW

- Must-have, Should-have, Could-have, Won’t-have
  - Cannot exist without - must
  - Possible to do it later - should
  - Delay until after release - could
  - Revisit at a later date - won’t
- Quick but error-prone - use when prioritization is part of an iterative process
- Subjective and biased to the loudest voice
- No model of time - not clear what release “must” or “won’t” belongs to
- Need some other way of mapping out how the product gets to v1.0 (MVP)

Kano

- Measure ‘satisfaction’ against completeness
- Also includes “indifferent” (customers don’t notice)
- Performance is where companies typically compete



More tips for practical requirements prioritization

- Maintain a list of requirements
- Record necessity interdependencies
- Annotate requirements by effort
- Annotate requirements by relative importance
- Do triage overtly (and involve the right stakeholders: customers, developers and financial representatives)
-

[Alan Davis, The Art of Requirements Triage, 2003]

- Establish a teamwork mentality (instead of adversarial relationships)
- Understand the optimistic, pessimistic and realistic approaches
- Plan more than one release at a time
- Replan before every new release
- Don't be intimidated into a solution
- Remember that perfection is impossible

In this lecture

- Discussed requirements prioritization as an important activity to release planning
- However difficult to achieve
- Introduced two methods for requirements prioritization
- The cost-value approach that uses the AHP method
- Outlined a number of practical steps when doing prioritization

References

Davis, A. The art of requirements triage, *IEEE Computer*, March 2003

Karlsson, J. and Ryan, K. A cost-value approach for prioritizing requirements, *IEEE Software*, Sept/Oct 1997

Regnell, B. *et al*, An industrial case study on distributed prioritization in market-driven requirements engineering for packaged software, *Requirements Engineering 6*, 51-62

Gralha, C. et al., The evolution of requirements practices in startups, Proceedings of the International Conference on Software Engineering, 2018