

CS 361 – Software Engineering

Requirements Prioritization

Kamonphop Srisopha
Churee Techawut



Faculty of Science, Chiang Mai University
คณะวิทยาศาสตร์ มหาวิทยาลัยเชียงใหม่

Agenda

- Requirements Prioritization
 - Six Prioritization Techniques
 - Multiple-Criteria Decision Analysis (MCDA)

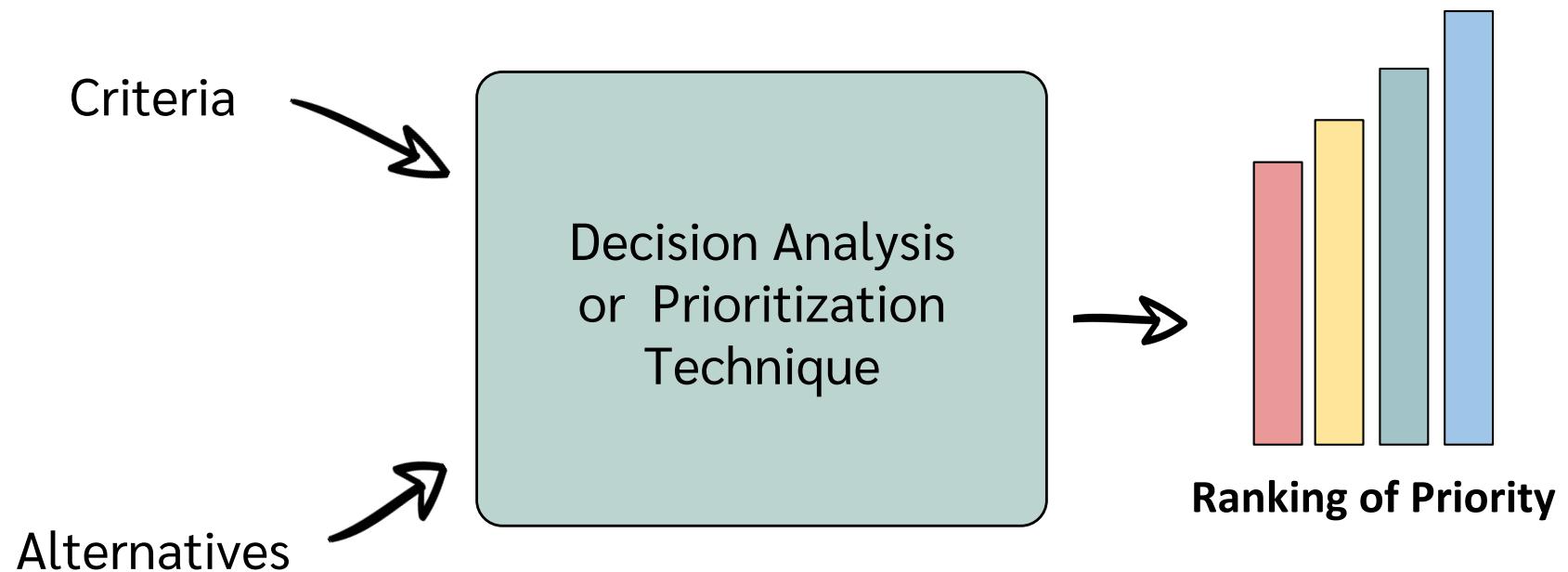
Requirements Prioritization

Requirements Prioritization

ถ้าเรานำสิ่งที่ไม่
จำเป็นเข้าก่อน
จะทำให้ไม่
สามารถใส่สิ่งที่
จำเป็นได้พ่อ



Requirements Prioritization



เรา Prioritize เพื่อว่าเรามีเวลา, กำลัง, หรือทรัพยากร ที่จะทำทุกอย่างได้หมด

Prioritization Techniques

MoSCow

Eisenhower Matrix

Kano Method

Minimal
marketable feature
set (MMFS)

Benefits vs Cost

Value-Based

MoSCow – Prioritization Techniques

เป็นเทคนิคการจัดความสำคัญแบบง่าย (มีแค่ 1 criterion) ซึ่งจะแบ่ง task ออกเป็น 4 กลุ่ม

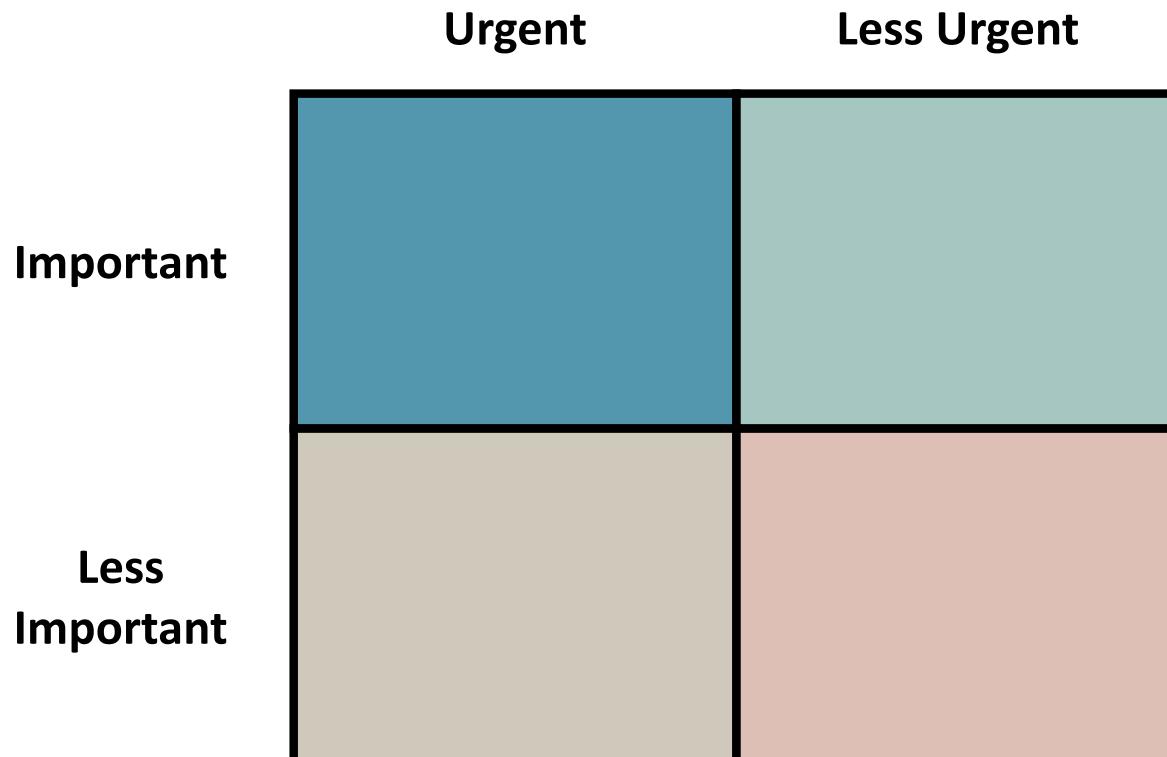
- Must have (จำเป็นต้องมี)
- Should have (ควรจะมี)
- Could have (มีก็ดีแต่ไม่มีก็ได้)
- Will not have or not right now (ไม่จำเป็น หรือ ยังไม่ใช่ตอนนี้)



ถ้าเป็นวัยว茫ในร่างกายคนเราจะจัดเรียง Priority โดยใช้ MoSCow Technique อย่างไร

Eisenhower Matrix

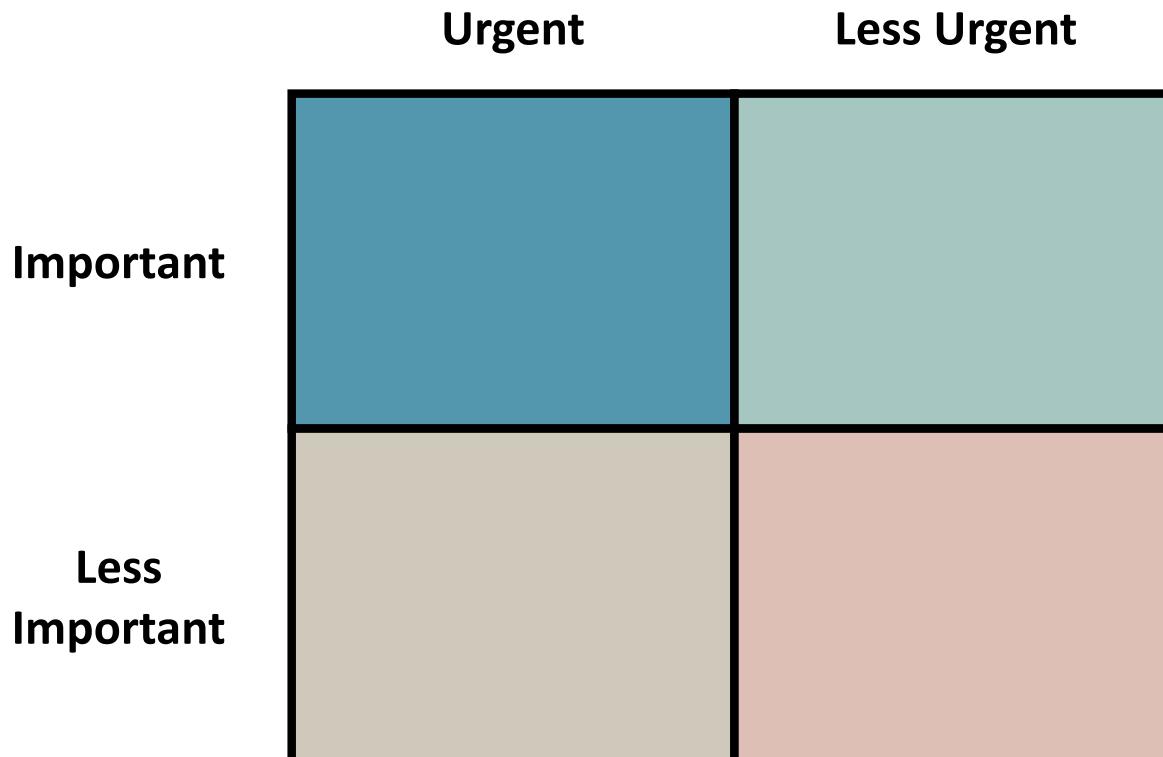
เรียกอีกอย่างว่า Urgent-Important Matrix คิดค้นโดย Dwight D. Eisenhower (ประธานาธิบดีสหรัฐคนที่ 34, 1953 - 1961) เป็นเทคนิคที่ช่วยในการจัดความสำคัญของงานต่างๆ โดย **2 Criteria** คือ 1) ระดับความเร่งด่วน (Urgency) และ 2) ความสำคัญ (Importance)



Eisenhower Matrix – A college student

Tasks:

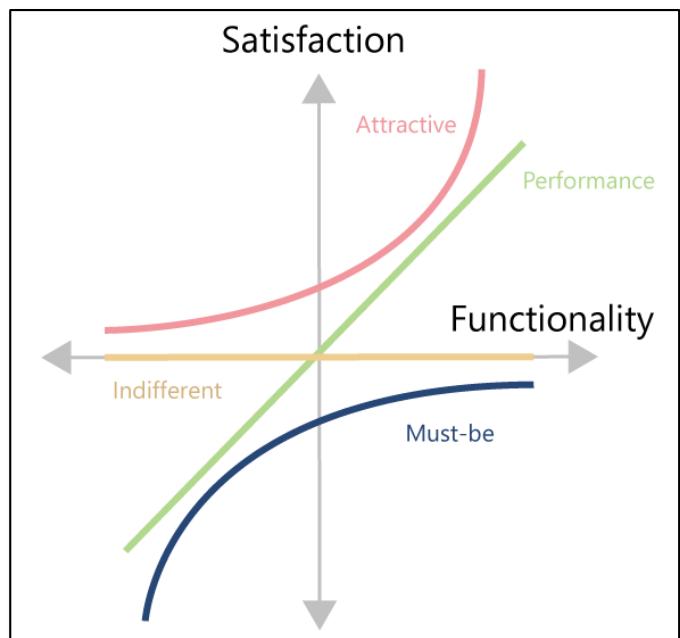
- ทำการบ้านส่งอาจารย์วิชา CS 361 ที่มี deadline ในวันพรุ่งนี้
- เตรียมตัวสอบมิดเทอมของวิชา CS 361
- ตอบ line ที่เพื่อนส่งมาชวนไปเมญ่าคืนนี้
- ดู series เกาะลึใน Netflix และเล่นเกมส์



Kano Method

เป็นเทคนิคการจัดความสำคัญของความต้องการของลูกค้า โดยจะจัดประเภทความต้องการตามระดับของฟังก์ชันการทำงานและระดับความพึงพอใจของลูกค้า พัฒนาโดย Dr. Noriaki Kano ชาวญี่ปุ่น

มี 4 กลุ่มหลักๆ:



- **Must-be Needs** (ความต้องการพื้นฐาน): ลูกค้าคาดหวังว่าจะได้รับ หากไม่ได้รับลูกค้าจะไม่พึงพอใจมาก
- **Performance Needs** (ความต้องการด้านประสิทธิภาพ): ยิ่งฟังก์ชันการทำงานมีประสิทธิภาพดีมากเท่าไร ลูกค้าก็จะพึงพอใจมากขึ้นเท่านั้น
- **Attractive Needs** (ความต้องการที่น่าประทับใจ): เกินความคาดหวังของลูกค้า การที่สามารถทำให้ลูกค้าตื่นเต้นและพึงพอใจเมื่อได้รับความต้องการนี้
- **Indifferent Needs** (ความต้องการที่ได้ก็ดีไม่ได้ก็ไม่เป็นไร): ไม่ส่งผลกระทบต่อความพึงพอใจของลูกค้าไม่ว่าฟังก์ชันของการทำงานจะอยู่ในระดับไหนก็ตาม

Kano Method - Example



Concert

Must-be Needs:

- ระบบเครื่องเสียง, ไฟ
- สถานที่ ที่นั่ง ที่ยืน ความจุคน
- การรักษาความปลอดภัย

Performance Needs:

- ลำโพงต้องฟังชัด เสียงใส ไร้เสียงสะท้อน
- ไฟเปลี่ยนสีตามเพลงตามจังหวะ
- นักร้องหรือวงที่มามีซื่อเสียง ร้องเพราะ

Attractive Needs:

- นักร้องรับเชิญที่ไม่ได้แจ้งให้ทราบล่วงหน้า
- เพลงพิเศษหลังจบการแสดง

Indifferent Needs:

- Design ของตัวเข้างาน
- สีของบูดของพนักงาน

Minimal Marketable Feature Set (MMFS)

Focus ความต้องการกับที่เป็นสิ่งจำเป็นสำหรับการเริ่มต้นการพัฒนาผลิตภัณฑ์หรือบริการ (**must-have functionality**) โดยกำหนดคุณสมบัติหรือความต้องการอย่างน้อยที่สุดที่จะสามารถให้คุณค่า* กับผู้ใช้ได้ (Value) หรือสามารถนำมาย用หรือนำออกสู่ตลาดได้โดย features เพียงแค่ใน set นั้น และจากนั้นใช้ set ความต้องการเหล่านี้เป็นพื้นฐานในการพัฒนาและเพิ่มสิ่งที่สำคัญอื่นๆขึ้นในอนาคต

เช่น ในการพัฒนา application สำหรับจัดการ Project

MMFS

1. User Registration and Authentication
2. Task Creation and Assignment
3. Task Progress Tracking and Reporting

Note:

*ขึ้นอยู่กับว่าผู้ใช้ต้องการคุณค่าอะไร

Benefits vs Cost

เทคนิคการจัดลำดับความสำคัญระหว่างประโยชน์กับต้นทุน (Benefits vs Cost prioritization) โดยพิจารณาประโยชน์ที่จะได้รับและต้นทุนที่ต้องใช้ในการดำเนินโครงการนั้นๆ นอกจากนี้ยังช่วยให้เราสามารถตัดสินใจเลือกโครงการหรือคุณสมบัติที่มีความคุ้มค่ามากที่สุดต่อต้นทุนที่มีอยู่



เช่น จอ Gaming Monitor ที่มีค่า response time (เวลาที่สีของ pixel เปลี่ยนจากดำเป็นขาว) อยู่ที่ 0.5 milliseconds (ms) แต่ นักศึกษาต้องการ 0.1ms แต่ถ้าจะ upgrade จาก 0.5 ไป 0.1 นักศึกษาต้องเพิ่มเงินถึง 5,000 บาท คำถามคือประโยชน์ที่ได้รับ จาก 0.1ms คุ้มค่ากับที่จะเสียเพิ่ม 5,000 หรือไม่?

Value-Based

จัดลำดับความสำคัญของสิ่งต่างๆตาม Criteria ที่ประเมินเป็นตัวเลขหรือคุณภาพ (Value) ได้ ซึ่ง Criteria พวgnี้เราสามารถเลือกขึ้นมาได้เอง ตามความเหมาะสม

เช่น เราอาจจะให้ความสำคัญกับ Criteria เหล่านี้

- **Business Value** (คุณค่าทางธุรกิจ) 1 *low* – 10 *high*
- **Relative Penalty** (ค่าผลกระทบหากไม่มีสิ่งนั้น) 1 *low* – 10 *high*
- **Implementation Cost** (ค่าต้นทุนในการทำสิ่งนั้น) 1 *low* – 10 *high*

เพราะฉะนั้นจากตัวอย่าง Criteria ด้านบน เราควรจะเลือกหาความต้องการที่
1) มีคุณค่าทางธุรกิจมาก 2) ผลกระทบมากหากไม่มีมัน และ 3) เสียต้นทุนในการการทำน้ำอย มาพัฒนา ก่อน

Value-Based

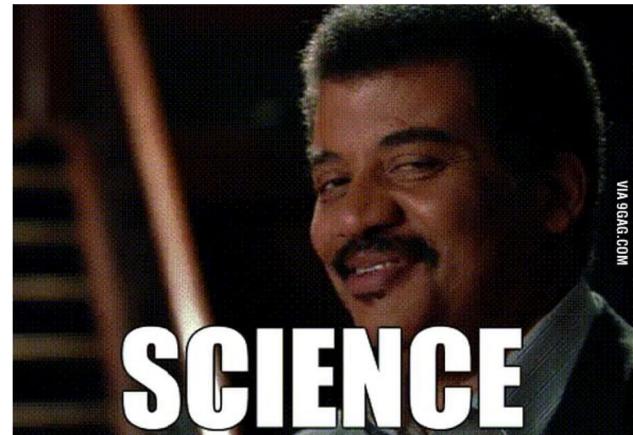


- ถ้าเรามี Criteria มากมายหลายปัจจัย และ มี Alternative มากมายที่ต้องนำมาคิด เราจะหาความสำคัญของ Alternative เหล่านั้นอย่างไร ?
- แล้วถ้าเราอยากให้ความสำคัญกับ Criterion หนึ่งมากกว่าอีก Criterion หนึ่งล่ะ?
- ถ้าผลกระทบของการเลือกทำในสิ่งหนึ่น ถ้าเลือกผิดมันมีมูลค่ามากล่ะ?
- เราจะเลือกอย่างเป็นกลาง และไม่ใช่ตามอำเภอใจหรือโดยสัมภានตญาณได้อย่างไร (Choose objectively and not subjectively or intuitively)?

Introducing ...

Multiple-Criteria Decision Analysis

(การวิเคราะห์เพื่อการตัดสินใจแบบหลายหลักเกณฑ์)



Multiple Criteria Decision Questions?

- ตัวอย่างของสิ่งที่นักศึกษาต้องตัดสินใจก่อนมาเรียนในวันนี้
(โดยต้องคำนึงถึงหลายหลักเกณฑ์หรือปัจจัย)
- ตัวอย่างของสิ่งที่ Software Engineer ต้องตัดสินใจ (โดยต้องคำนึงถึงหลายหลักเกณฑ์หรือปัจจัย)

หน้าตาของ MCDA Matrix

A set of criteria (เกณฑ์) ความสำคัญของ Criteria (รวม 100%)

ค่า Ranking

Overall Score Alternatives

| Overall Score | Alternatives | Criteria | | | | | | |
|---------------|---------------|-------------|-------------|-------------|-----|-------------|----------|-----|
| | | Criterion 1 | Criterion 2 | Criterion 3 | ... | Criterion N | | |
| | | W_1 | W_2 | W_3 | ... | W_N | | |
| $S(A_1)$ | | | | | | | S_{1N} | |
| $S(A_2)$ | | | | | | | S_{2N} | |
| $S(A_3)$ | | | | | | | S_{3N} | |
| ... | ... | ... | ... | ... | ... | ... | ... | ... |
| $S(A_N)$ | Alternative N | S_{N1} | S_{N2} | S_{N3} | ... | | S_{NN} | |

แล้วค่า $S[i,j]$ จะหาออกมาได้อย่างไร ?

A set of alternatives (ทางเลือก)

ค่า score $S[i,j]$ คือค่าประเมินว่า alternative i มีคุณค่า (value) เท่าไหร่กับ criterion j

Estimation Exercises



อายุเฉลี่ยของยีราฟ (ห้ามใช้อุปกรณ์ Electronics)



ผู้สอนมีความสูงเท่าไหร่ (ถามผู้สอนไม่ได้)



ค่าใช้จ่ายถ้าจะเดินทางไปเที่ยวประเทศ Switzerland

Techniques for Estimating

Expert Opinion

เป็นการประมาณค่าโดยอาศัยความรู้และประสบการณ์ของผู้เชี่ยวชาญ เช่น การประมาณระยะเวลาในการพัฒนาระบบ login ของธนาคาร โดยขอความคิดเห็นจากผู้เชี่ยวชาญที่มีประสบการณ์ในการพัฒนาความต้องการนี้มาก่อน

Analogy

เป็นการประมาณค่าโดยเปรียบเทียบกับโครงการหรืองานที่คล้ายคลึงกัน เช่น ดูราคาเช่าคอนโดที่ตั้งอยู่ในพื้นที่เดียวกันเพื่อประมาณราคาเช่าคอนโดที่ต้องการ

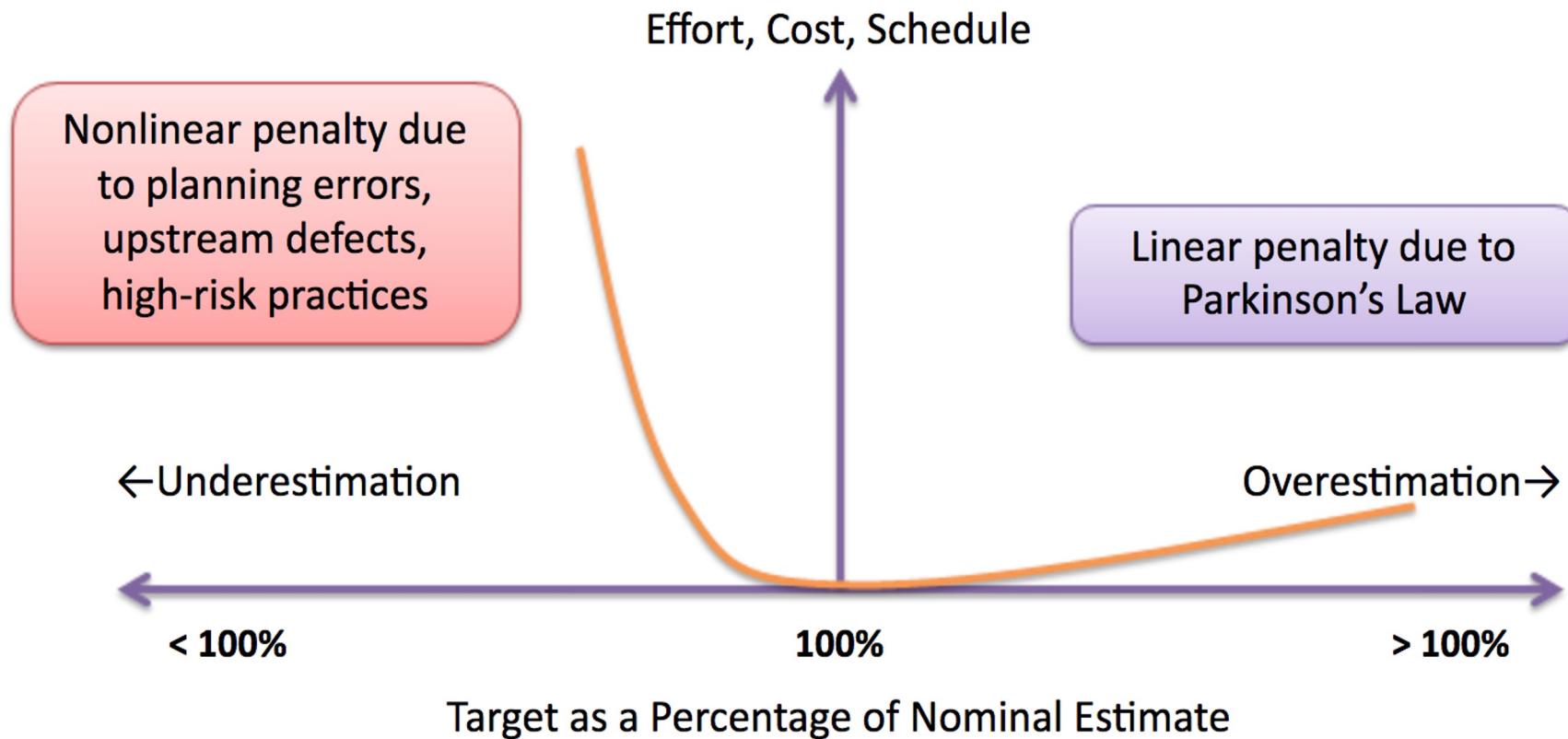
Disaggregation

เป็นการประมาณค่าโดยแยกเป็นส่วนย่อย ๆ และประมาณค่าในแต่ละส่วน จากนั้นนำค่าที่ได้มาบวกกันเพื่อหาค่าประมาณทั้งหมด เช่น งบรายนต์โดยแยกเป็นราคากองรถ, ภาษี, ค่าจดทะเบียน, ค่าประกัน และค่าบำรุงรักษา เสร็จแล้วนำมารวบกันเพื่อหาค่าใช้จ่ายทั้งหมด



การประมาณค่าด้วยเทคนิคต่าง ๆ นี้ จะช่วยให้เราสามารถวางแผน และตัดสินใจได้ดีขึ้น อย่างไรก็ตาม ควรใช้หลายเทคนิคร่วมกันเพื่อควบคุมความคลาดเคลื่อน และเพิ่มความมั่นใจในการประมาณค่า

Over-Estimation vs Under-Estimation



Penalties of underestimation more severe than those for overestimation. If you can't estimate with complete accuracy, it's better to err on the side of overestimation

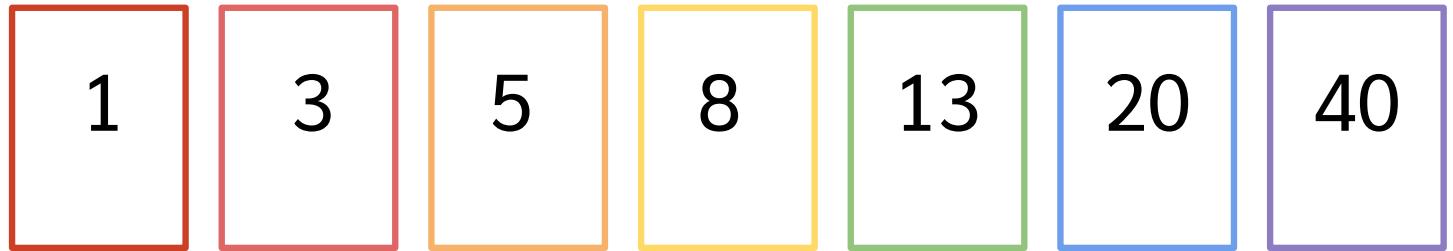
– Steve McConnell



แต่ทีม SE มีหนึ่งๆ มีสมาชิกหลายคน และ
อย่างนี้จะ Estimate ค่าต่างๆยังไงดีหละ
ควรให้คนเดียว Estimate เลยดีมั้ย?

Agile/Scrum Planning Poker

How to play:

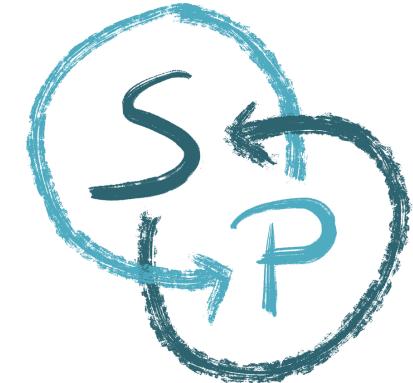


1. ทุกคนในทีมจะได้รับการ์ดที่มีตัวเลขจากน้อยไปมาก (อาจใช้ตัวเลขจาก ลำดับ Fibonacci)
2. Project Manager (หรือ Facilitator) จะเลือกความต้องการและอธิบายความต้องการพอ สั้งเขป พร้อมทั้งอธิบายตัวเลขในการ์ดว่าหมายความว่าอะไร เช่น จำนวนชั่วโมงทำงาน, จำนวนวัน, ความยาก, ฯลฯ
3. สมาชิกในทีมแต่ละคนจะเลือกการ์ดที่ตรงกับปริมาณที่เข้าคาดไว้
4. เมื่อทุกคนเลือกเสร็จ Project Manager จะให้ทุกคนหงายการ์ด จะได้เห็นว่าการประเมิน ความต้องการนั้นๆของแต่ละคนแตกต่างกันอย่างไร
5. ถ้ามีการประเมินที่แตกต่างกันมาก ทีมจะสนทนาเพื่อทำความเข้าใจว่าทำไมมีความ คิดเห็นที่แตกต่าง
6. ทำซ้ำจากข้อ 3 จนกว่าทุกคนจะเข้าใจไปในทางเดียวกัน และเห็นด้วยกันกับค่าประเมินที่ ออกมา

Agile/Scrum Planning Poker

Free Online Tool for Planning Poker:

- <https://planningpokeronline.com/>
- <https://www.planitpoker.com/>
- <https://www.scrumpoker-online.org/en/>



A screenshot of a web-based planning poker application. At the top left is a team icon labeled "Death Star Team" with a "Voting: Add CTA" button. At the top right is a user profile for "David" with an "Invite players" button. Below the header, several team members are shown with their names and profile pictures: Sara, Michael, Jennifer, and another member whose card is currently being voted on, indicated by a blue box containing the text "Waiting for player's votes...". At the bottom, a row of numbered cards is displayed: 0, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, and ?. A small robot icon is in the bottom left corner.



หน้าตาของ MCDA Matrix

A set of criteria (เกณฑ์) ความสำคัญของ Criteria (รวม 100%)

ค่า Ranking

Overall Score Alternatives

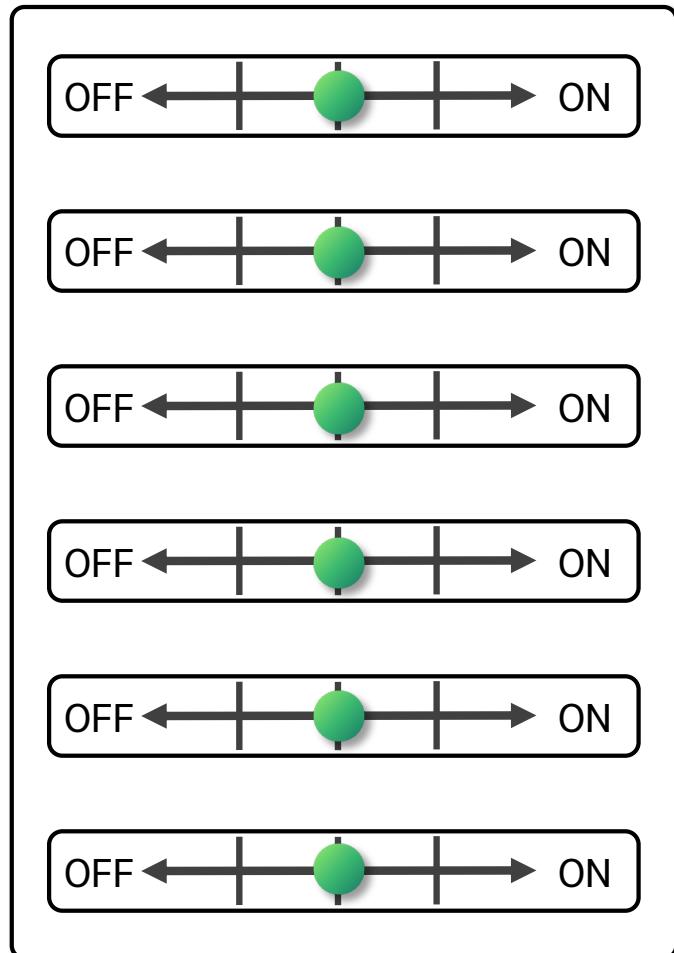
| Overall Score | Alternatives | Criteria | | | | | | |
|---------------|---------------|-------------|-------------|-------------|-----|-------------|----------|-----|
| | | Criterion 1 | Criterion 2 | Criterion 3 | ... | Criterion N | | |
| | | W_1 | W_2 | W_3 | ... | W_N | | |
| $S(A_1)$ | | | | | | | S_{1N} | |
| $S(A_2)$ | | | | | | | S_{2N} | |
| $S(A_3)$ | | | | | | | S_{3N} | |
| ... | ... | ... | ... | ... | ... | ... | ... | ... |
| $S(A_N)$ | Alternative N | S_{N1} | S_{N2} | S_{N3} | ... | | S_{NN} | |

ค่าความสำคัญของ Criteria ควรจะจัดการอย่างไร ?

A set of alternatives (ทางเลือก)

ค่า score $S[i,j]$ คือค่าประเมินว่า alternative i มีคุณค่า (value) เท่าไหร่กับ criterion j

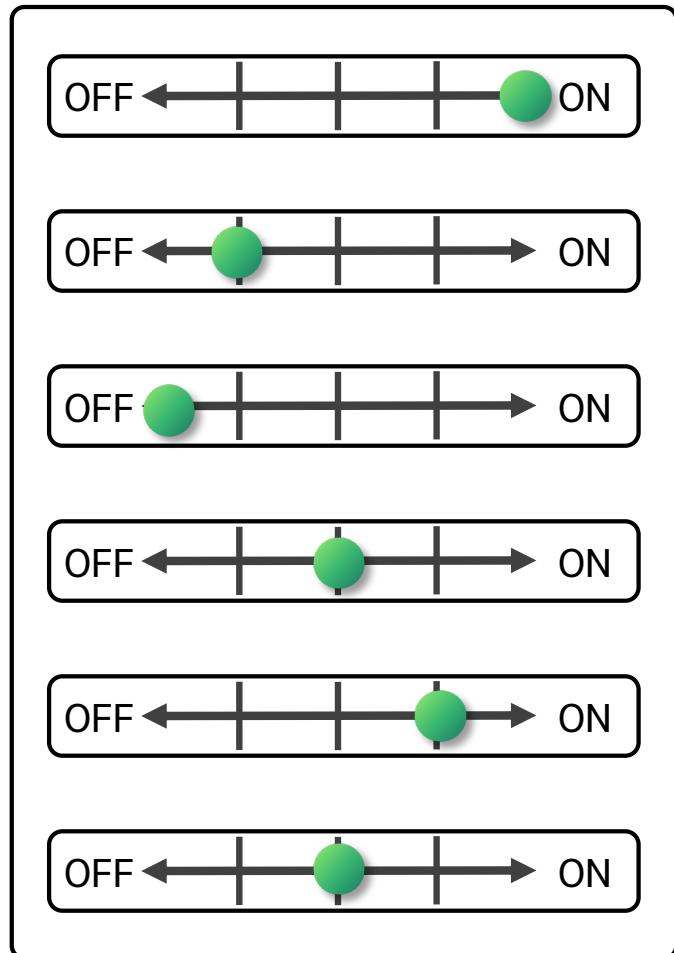
Project Success/Trade-off Slider



0 25 50 75 100
1 2 3 4 5

Project Success/Trade-off Slider ของโครงการในแต่ criterion จะเริ่มที่ค่า 3 (กลาง) ในช่วงระหว่าง 1 ถึง 5 โดยทีมผู้พัฒนา (รวมถึงผู้ที่มีส่วนได้ส่วนเสียในตัวโครงการ) จะเลื่อนตัว slider ขึ้นหรือลง เพื่อสะท้อนถึงการผสานปัจจัยที่เหมาะสมในการกำหนดความสำเร็จของโครงการ (success criteria) แต่อย่างไรตามจะมีกฎอยู่ว่า การเลื่อนขึ้นทุกครั้งต้องมีการปรับลดลงที่เท่ากัน (Trade-offs)

Project Success/Trade-off Slider

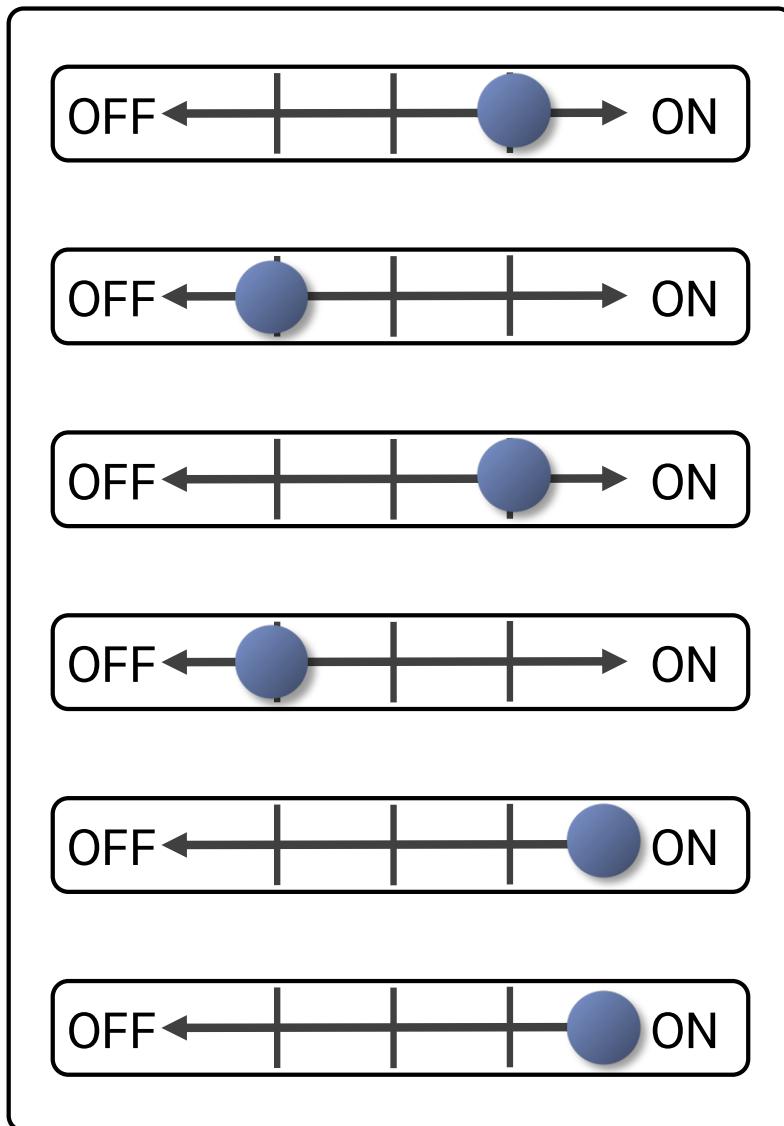


0 25 50 75 100
1 2 3 4 5

Project Success/Trade-off Slider ของโครงการในแต่ criterion จะเริ่มที่ค่า 3 (ค่ากลาง) ในช่วงระหว่าง 1 ถึง 5 โดยทีมผู้พัฒนา (รวมถึงผู้ที่มีส่วนได้ส่วนเสียในตัวโครงการ) จะเลื่อนตัว slider ขึ้นหรือลง เพื่อสะท้อนถึงการผสมผสานของปัจจัยที่เหมาะสมในการกำหนดความสำเร็จของโครงการ (success criteria) แต่อย่างไรตามจะมีกฎอยู่ว่า การเลื่อนขึ้นทุกครั้งต้องมีการปรับลดลงที่เท่ากัน (Trade-offs)

Project Success/Trade-off Slider

ตัวอย่าง Criteria



ความพึงพอใจของผู้มีส่วนได้ส่วนเสีย

ส่งมอบตามเวลาที่กำหนด

 ตามหลักแล้ว
แบบนี้เป็นไปได้หรือไม่?

ตรงตามข้อกำหนดคุณภาพ

ความพึงพอใจของทีมพัฒนา

หน้าตาของ MCDA Matrix

A set of criteria (เกณฑ์) ความสำคัญของ Criteria (รวม 100%)

ค่า Ranking

Overall Score Alternatives

| Overall Score | Alternatives | Criteria | | | | | | |
|--------------------|---|--|-----------------|-----------------|-----|-------------|-----|-----------------|
| | | Criterion 1 | Criterion 2 | Criterion 3 | ... | Criterion N | | |
| S(A ₁) |  | เมื่อมีข้อมูลครบแล้ว จะหาค่า Ranking ออกมายังไง? | | | | | | S _{1N} |
| S(A ₂) | | | | | | | | S _{2N} |
| S(A ₃) | | | | | | | | S _{3N} |
| ... | ... | ... | ... | ... | ... | ... | ... | ... |
| S(A _N) | Alternative N | S _{N1} | S _{N2} | S _{N3} | ... | | | S _{NN} |

A set of alternatives (ทางเลือก)

ค่า score $S[i,j]$ คือค่าประเมินว่า alternative i มีคุณค่า (value) เท่าไรกับ criterion j

Two Widely-Used MCDA Techniques

Simple
Additive
Weighting

TOPSIS

Simple Additive Weighting

$S(A_i)$ can be calculated from

$$S(A_i) = \sum_{j=1}^n W_j S_{ij}$$

| Overall Score | Alternatives | Criteria | Criterion 1 | Criterion 2 | Criterion 3 | ... | Criterion N |
|---------------|---------------|----------|-------------|-------------|-------------|-----|-------------|
| | | | W_1 | W_2 | W_3 | ... | W_N |
| $S(A_1)$ | Alternative 1 | | S_{11} | S_{12} | S_{13} | ... | S_{1N} |
| $S(A_2)$ | Alternative 2 | | S_{21} | S_{22} | S_{23} | ... | S_{2N} |

For example:

$$S(A_1) = W_1 * S_{11} + W_2 * S_{12} + W_3 * S_{13} + \dots + W_N * S_{1N}$$

เสร็จแล้วเรียงลำดับ alternative ตามค่า S

TOPSIS

ย่อมาจาก **Technique for Order Preference by Similarity to Ideal Solution** (Hwang & Yoon, 1981)

Key Idea: กำหนดกลุ่มเป้าหมายอุดมคติเชิงบวก (positive-ideal solution) หรือเป้าหมายที่มีค่าที่ดีที่สุดในแต่ละเกณฑ์ และกลุ่มเป้าหมายอุดมคติเชิงลบ (negative-ideal solution) หรือเป้าหมายที่มีค่าที่แย่ที่สุดในแต่ละเกณฑ์ แล้วหา alternative ที่อยู่ใกล้เป้าหมายเชิงบวกและห่างจากเป้าหมายเชิงลบที่สุด

How to do TOPSIS

- 1 . สร้าง Decision Matrix
- 2 . Normalize Decision Matrix ในแต่ละ Criteria
- 3 . นำค่า Weight ของแต่ละ Criteria เข้าไปคูณ
- 4 . หา Positive Ideal Solution
- 5 . หา Negative Ideal Solution
6. หา Distance ระหว่างแต่ละ Alternative กับ Positive และ Negative Ideal solutions
7. หาค่า closeness ของแต่ละ alternative จากค่า distance ในข้อ 6
8. นำค่าในข้อ 7 มาจัดลำดับ alternative (rank แรกคือ alternative ที่ใกล้กับ positive และ ไกลจาก negative solutions มากที่สุด

TOPSIS – ตัวอย่าง with Excel

Decision Matrix

| Alternative | RAM | Memory | CPU speed | Weight | Looks |
|-------------|-----|--------|-----------|--------|-----------|
| | 0.2 | 0.2 | 0.2 | 0.3 | 0.1 |
| Laptop1 | 8 | 256 | 3.14 | 6.3 | Excellent |
| Laptop2 | 12 | 512 | 3.7 | 6.7 | Average |
| Laptop3 | 16 | 256 | 4.1 | 6.5 | Bad |



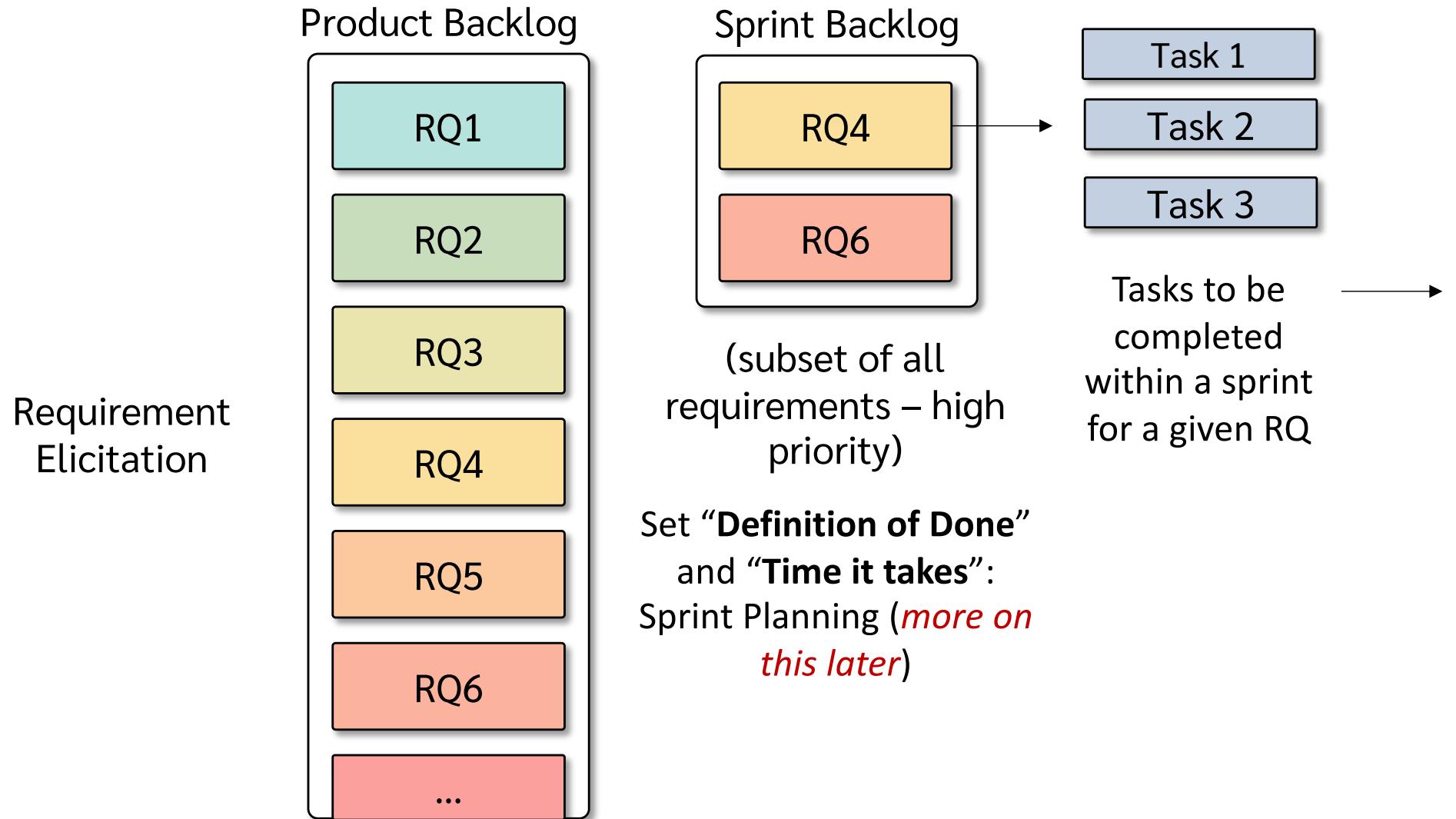
<https://cmu.to/topsis-excel>

TOPSIS – ตัวอย่าง with Python

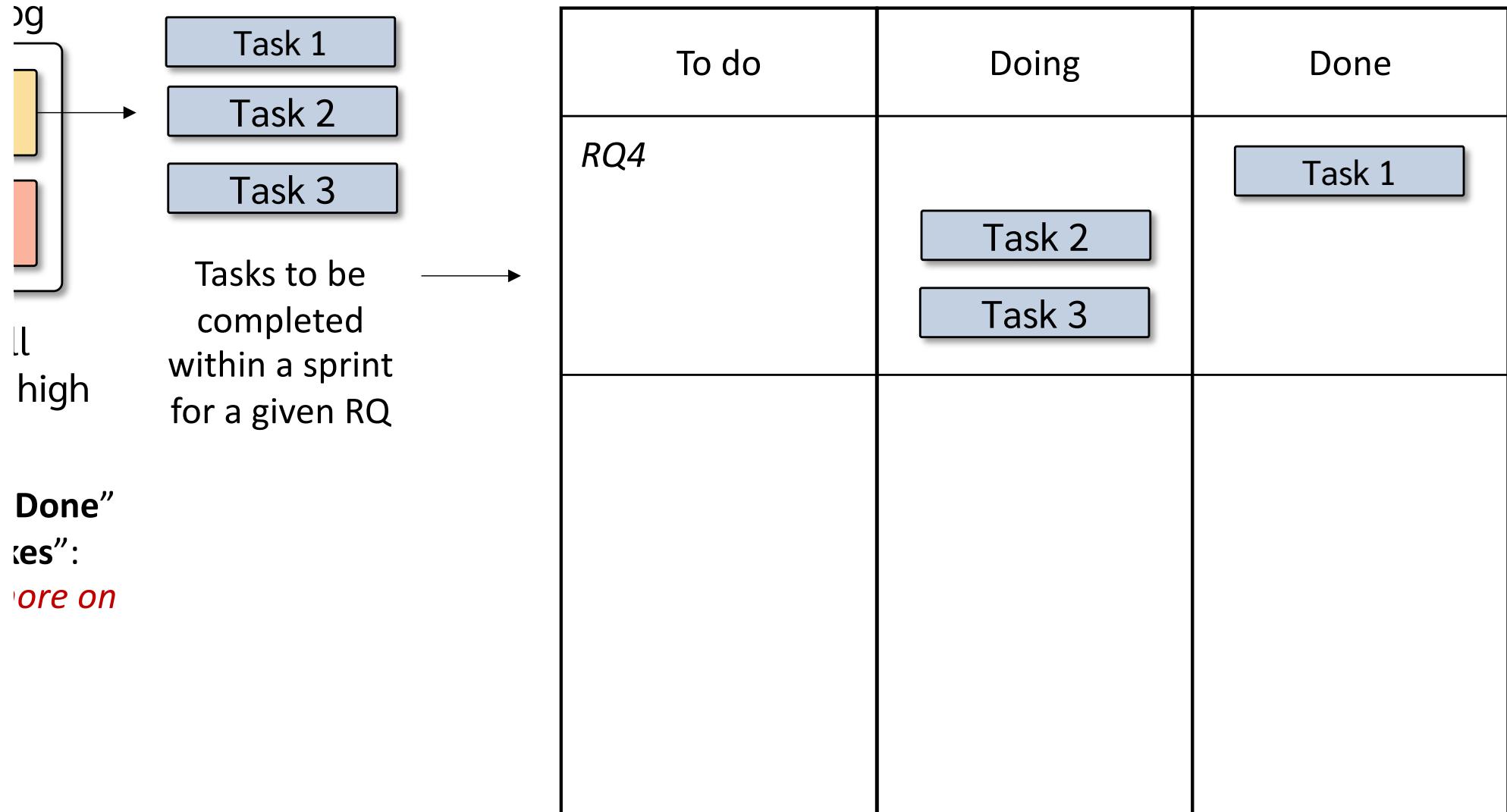


<https://cmu.to/t7Vgd>

Putting it All Together



Putting it All Together



Should be updated at least every day/daily Scrum

Questions?

