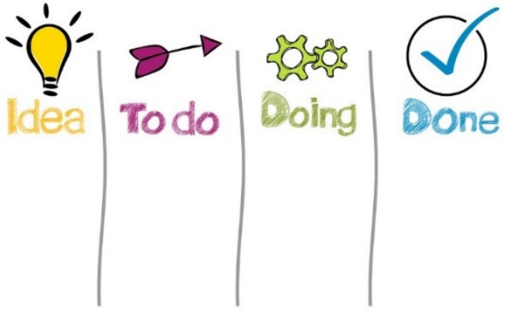


VO Softwareentwicklungsprozess

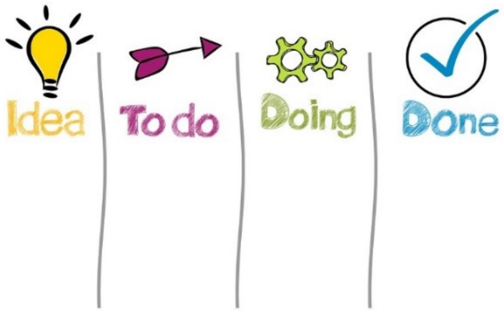
https://youtu.be/FbhcJXLiklo

Alexander Felfernig und Trang Tran

Institut für Softwaretechnologie, Inffeldgasse 16b/2

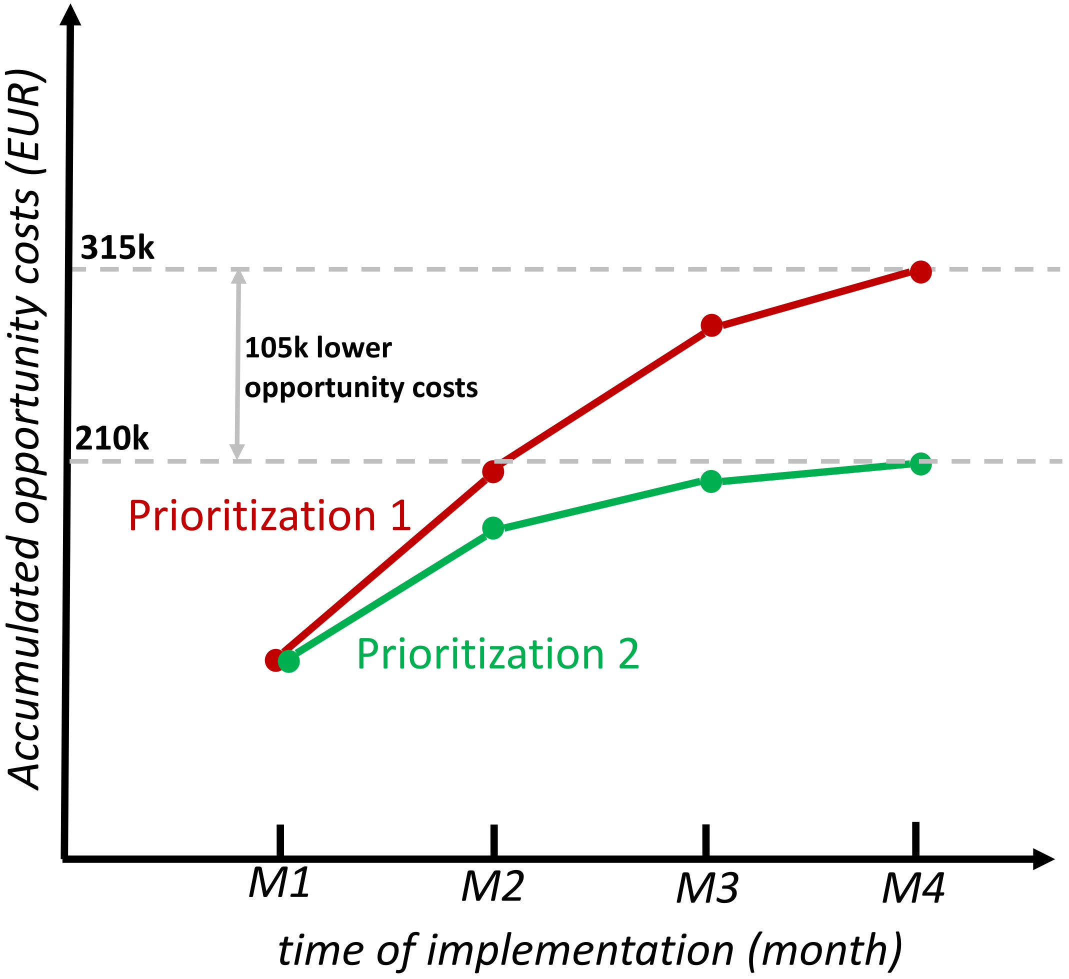
- Requirements Prioritization -

Requirements Prioritization in Software Projects



- Limited resources
- Dependencies between requirements
- Conflicting interests of stakeholders (customers, developers, and management)
- Different degree of market relevance of requirements
- **Task**: find an ordering of requirements such that the most relevant ones are implemented first

Why Requirements Prioritization?

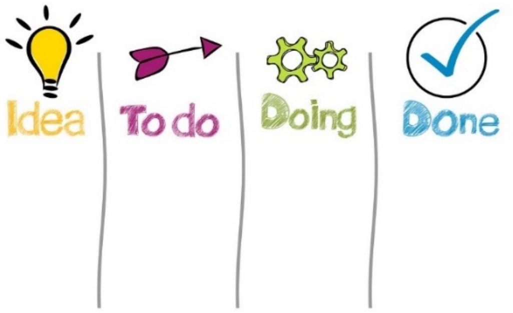


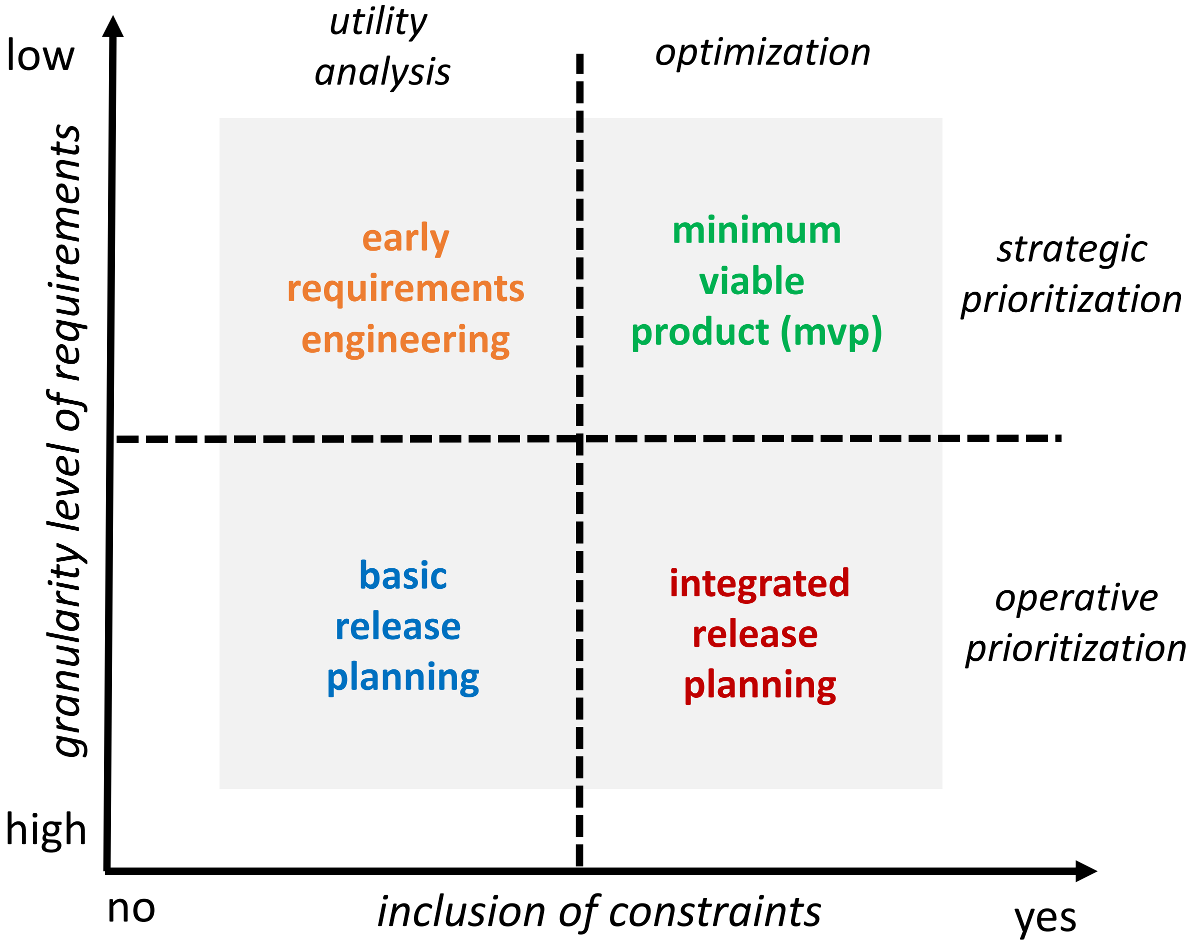
Requirement	Process improvements per month (EUR)
Req1	5k
Req2	20k
Req3	50k
Req4	30k

ID	Prioritization	Opportunity costs (M1..M4)
1	Req1, Req2, Req3, Req4	(5+20+50+30) + (20+50+30) + (50+30) + (30)=315k
2	Req4, Req3, Req2, Req41	(30+50+20+5) + (50+20+5) + (20+5) + (5)=210k

<u>Assumption</u>: implementation of one requirement / month.

Requirements Prioritization: Classification





utility analysis

optimization

early requirements engineering

minimum viable product (mvp)

strategic prioritization

basic release planning

integrated release planning

operative prioritization

low

high

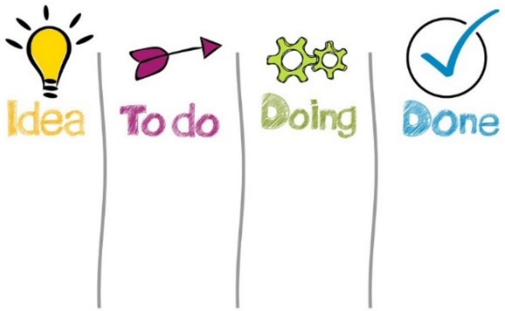
granularity level of requirements

no

inclusion of constraints

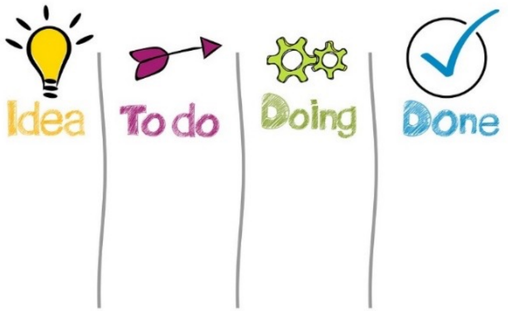
yes

- Early RE: estimating the utility of individual high-level features
- MVP: figuring out and starting with the most relevant features
- Basic release planning: detailed requirements, preferences regarding releases, no view on resources
- Integrated release planning: detailed requirements, dependencies, resources (e.g., other releases, tasks, etc.)



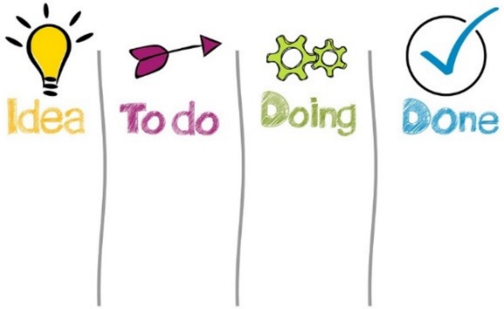
I. Strategic Prioritization

Early Requirements Engineering (RE)



- Goal: evaluating high-level requirements (features) w.r.t. relevance for the market or specific customer communities
- High level requirements (features), for example, *„the new e-learning software should include a motivation functionality that persuades users to intensively learn different course topics“*
- No specific technical or resource-related constraints are defined and taken into account

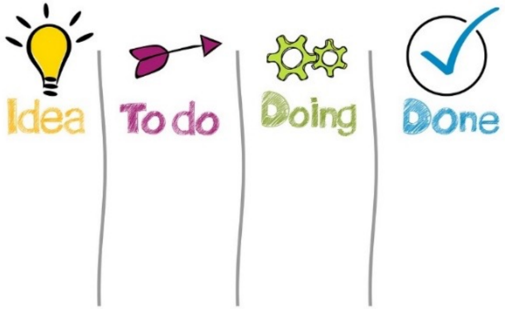
Example Requirements (Early RE)



Example high-level requirements (features) of an **e-learning software**…

Requirement	Requirement Description („e-learning“ Software)
Req1	Inclusion of a motivation feature that persuades users to intensively learn different course topics.
Req2	Inclusion of a natural language understanding feature that manages to automatically estimate an answer to a question.
Req3	Inclusion of an emotion detection feature that helps to interpret a user‘s current emotional state and includes this knowledge in learning item recommendation.
Req4	Inclusion of a recommendation feature that helps user to prioritize their learning efforts and to figure out the next learning item.

Prioritization Example (Early RE)



Stakeholder 1

Requirement	Effort	Profit	Risk
Req1	2	4	8
Req2	4	5	8
Req3	7	10	3
Req4	3	5	6

Stakeholder 2

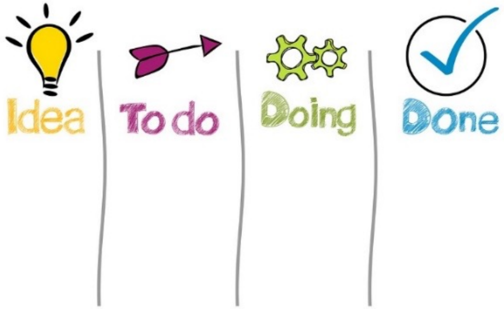
Requirement	Effort	Profit	Risk
Req1	3	3	7
Req2	4	4	7
Req3	8	9	2
Req4	4	5	7

Total

Require-ment	Effort(e)	Profit(p)	Risk(r)	utility(Reqi)	Rank
Req1	2.5	3.5	7.5	$\frac{2.5+3.5+7.5}{3}=4.5$	4
Req2	4.0	4.5	7.5	$\frac{4.0+4.5+7.5}{3}=5.33$	2
Req3	7.5	9.5	2.5	$\frac{7.5+9.5+2.5}{3}=\mathbf{6.5}$	**1**
Req4	3.5	5.0	6.5	$\frac{3.5+5.0+6.5}{3}=5.0$	3

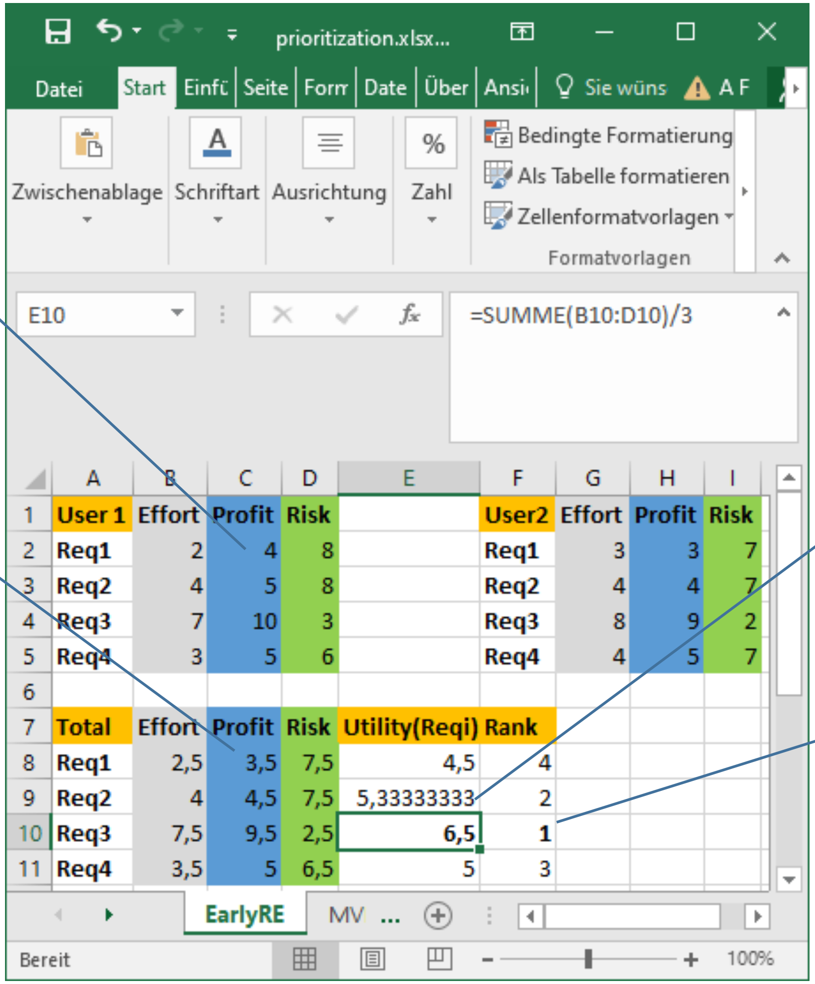
- Utility-based prioritization (ranking) of requirements
- Each stakeholder evaluates Reqi w.r.t. dimensions effort, profit, risk
- Aggregated values (AVG) are used for ranking
- $utility(Req) = \frac{e(Req)+p(Req)+r(Req)}{3}$

Prioritization Example contd. (MS Excel)



user-individual evaluation of Reqi w.r.t. the dimensions effort, profit, risk

aggregated evaluations



E10 =SUMME(B10:D10)/3

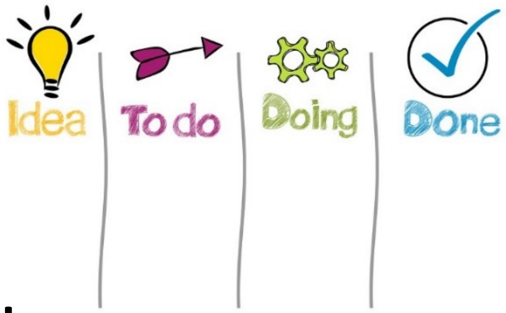
	A	B	C	D	E	F	G	H	I
1	User 1	Effort	Profit	Risk		User2	Effort	Profit	Risk
2	Req1	2	4	8		Req1	3	3	7
3	Req2	4	5	8		Req2	4	4	7
4	Req3	7	10	3		Req3	8	9	2
5	Req4	3	5	6		Req4	4	5	7
6									
7	Total	Effort	Profit	Risk	Utility(Reqi)	Rank			
8	Req1	2,5	3,5	7,5	4,5	4			
9	Req2	4	4,5	7,5	5,33333333	2			
10	Req3	7,5	9,5	2,5	6,5	1			
11	Req4	3,5	5	6,5	5	3			

utilities of individual requirements Reqi

proposed ranking (prioritization)

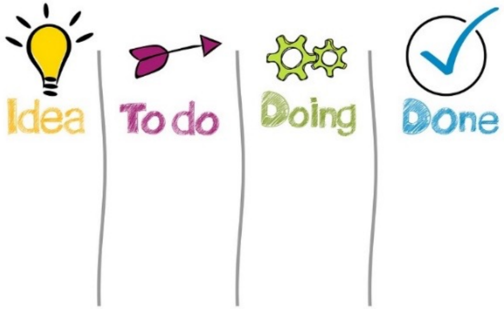
Remark: this .xlsx is provided in TeachCenter

Minimum Viable Product (MVP)



- Goal: figuring out the minimal set of features that should be part of a fully operable software that can be productively used by customers
- More fine-granular requirements, for example, *„the motivation features of the e-learning software should solely take into account the aspect of social influence“*
- Specifically, resource-related constraints are taken into account (e.g., availability of personnel)

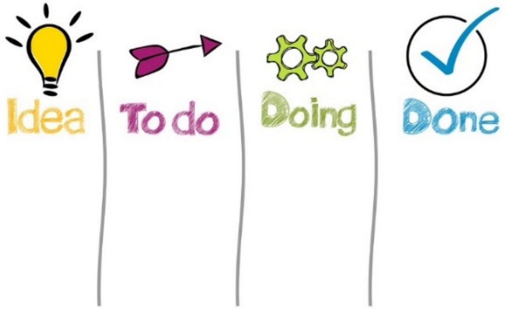
Example Requirements (MVP)



Example high-level requirements (features)
regarding an **e-learning software**…

Requirement	Requirement Description („e-learning“ Software)
Req1	Inclusion of a motivation feature that persuades users to intensively learn different course topics. For the first version, this feature shall generate a text message based social factors (*users who successfully completed the course, took a look at the following learning items*).
Req2	Inclusion of a natural language understanding feature that manages to automatically estimate an answer to a question. For the first version, this feature shall be based on the similarity between pre-specified keywords and the answer given by the user.
Req3	Inclusion of an emotion detection feature that helps to interpret a user's current emotional state and includes this knowledge in learning item recommendation. For the first version, emotion detection shall simply be based on emoticons selected by the user.
Req4	Inclusion of a recommendation feature that helps users to prioritize their learning efforts and to figure out the next learning item. For the first version, this feature shall be based on collaborative filtering, i.e., a basic match estimate between learning items and users.

Prioritization Example (MVP)

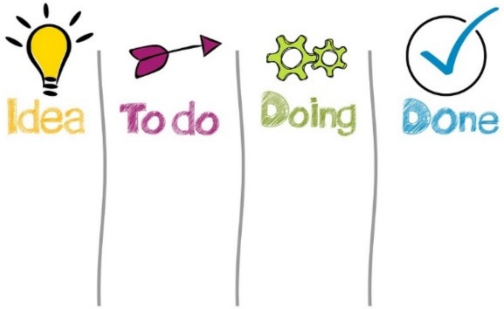


Requirement	Req1	Req2	Req3	Req4
utility(Reqi)	4.5	5.33	6.5	5.0
time(Reqi)	2.5	4.0	7.5	3.5
selected	1	1	0	1

- Optimization-based prioritization
- Example: maxtime = 10
- Goal: maximize overall utility
- Outcome: Req1,2,4 → 14.83

- Time limit: time(Req1)*selected(Req1)* ..
  * time(Req4)*selected(Req4) <= maxtime
- Utility maximization: utility(Req1)*selected(Req1)* ..
  *utility(Req4)*selected(Req4) → MAX

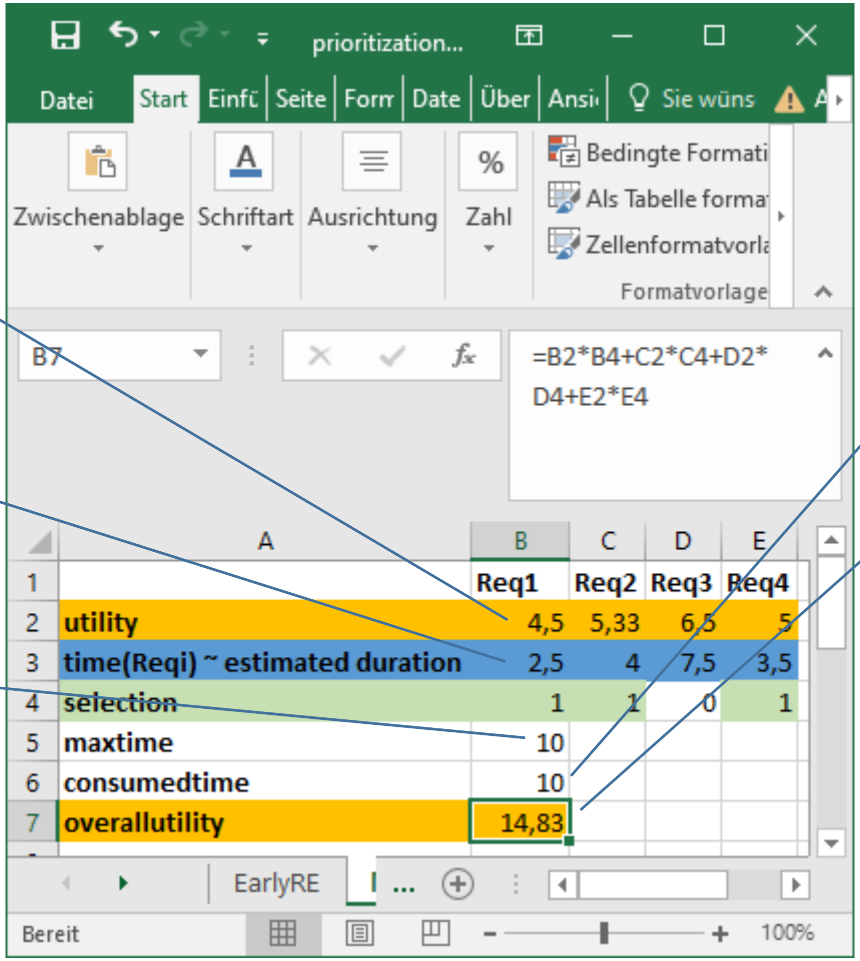
Prioritization Example contd. (MS Excel Solver)



evaluation of Reqi w.r.t. the dimensions effort, profit, risk

estimated time needed to implement Reqi

maximum time available to implement the MVP



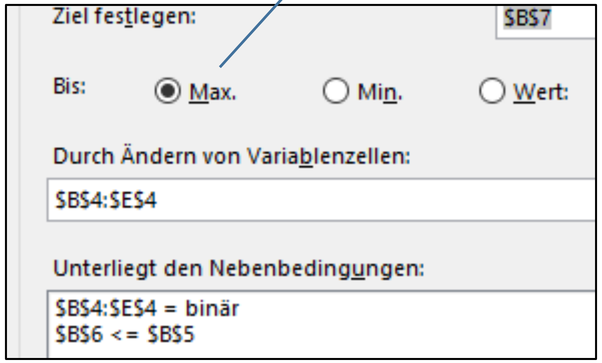
B7 =B2*B4+C2*C4+D2*D4+E2*E4

	A	B	C	D	E
1		Req1	Req2	Req3	Req4
2	utility	4,5	5,33	6,5	5
3	time(Reqi) ~ estimated duration	2,5	4	7,5	3,5
4	selection	1	1	0	1
5	maxtime	10			
6	consumedtime	10			
7	overallutility	14,83			

time needed to implement selected requirements

overall utility of requirements selected for MVP

maximize the overall utility



Ziel festlegen: $B$7

Bis: ● Max. ○ Min. ○ Wert:

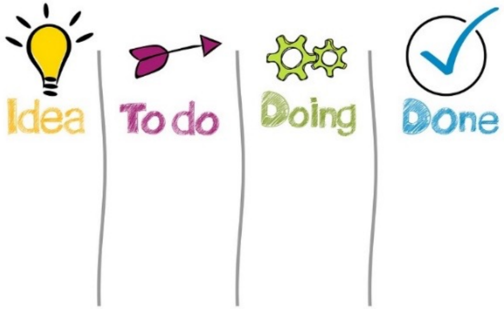
Durch Ändern von Variablenzellen:

$B$4:$E$4

Unterliegt den Nebenbedingungen:

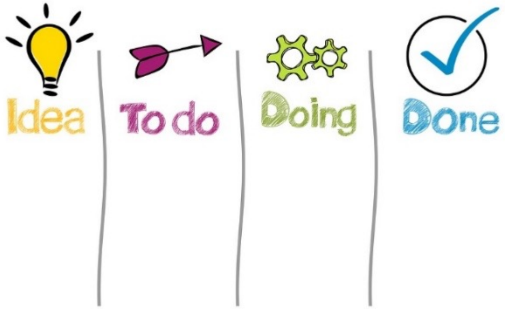
$B$4:$E$4 = binär
$B$6 <= $B$5

Remark: this .xlsx is provided in TeachCenter



II. Operative Prioritization

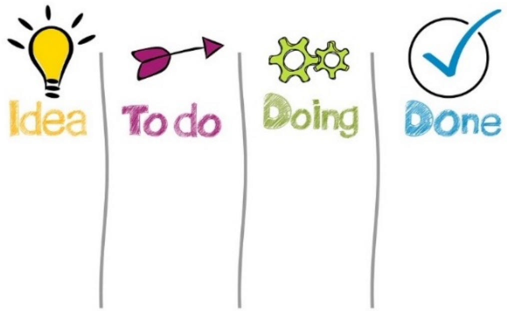
Example Requirements (Low-Level)



Example requirements regarding
a **Conference Timer** …

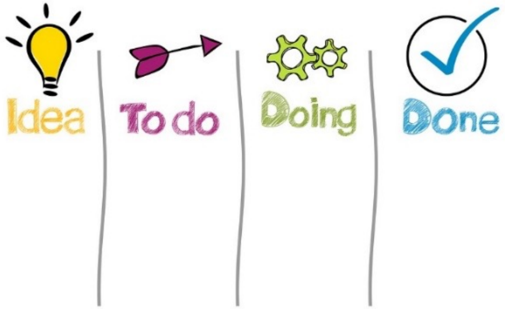
Requirement	Requirement Description („Conference Timer“ App)
Req1	The current time shall be shown in the chosen default setting (analog numbers).
Req2	The conference timer shall support the setting of current date and time (date, hh:mm:ss).
Req3	A mood feature shall support the selection of a clock mode (one out of „blue ocean“, „arctic coolness“, and „scientific quality“.
Req4	The conference timer function shall support the definition of a conference schedule, i.e., which speaker should do from when until when his/her presentation.
Req5	For each speaker, the conference timer shall support the execution of the defined schedule, i.e., starting the schedule and displaying the remaining available time.
Req6	Speakers shall be notified via “beep” function and visual signal (red blinking timer) 2 minutes before the of the presentation slot.
Req7	After the end of the current presentation, the next speaker should be announced, i.e., the timer shall display the speaker surname, presentation title, and starting time of the talk.

Basic Release Planning



- Goal: prioritization of the requirements to be implemented in the next *n releases* (special case: n=1, the *next-release problem*)
- Fine-granular requirements, for example, „*the user interface of the motivation function should show the recommended learning items and a 2-line persuasive message based on the typical learning behavior a user‘s social surrounding*“
- Often, dependency constraints are taken into account (e.g., requirement A should be implemented before requirement B)

Prioritization Example (Basic Release Planning)

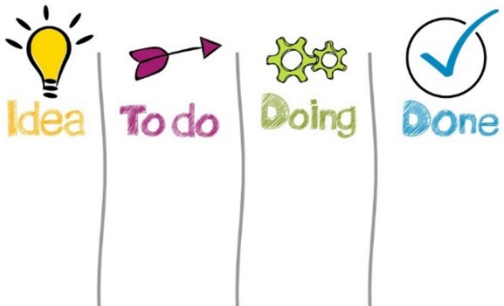


Requirement	User1	User2	User3	User4
Req1	1	2	1	1
Req2	2	1	1	1
Req3	3	1	2	1
Req4	3	3	3	2
Req5	1	2	4	2
Req6	2	2	2	2
Req7	2	2	2	3

- ur_{ij} = preferred release assignment of requirement j defined by user i
- for example: $ur_{17} = 2$
- ur'_{ij} = adaptation of ur_{ij}
- m = #users
- n = #requirements

- Consensus is needed: $\forall k$: $ur'_{1k} = ur'_{2k} = .. = ur'_{mk}$ (k=1..n)
- Adaptation per *user* x *requirement* pair: $chg_{ij} = |ur_{ij} - ur'_{ij}|$
- Minimize adaptations: $\sum_{i=1..m} \sum_{j=1..n} chg_{ij} \rightarrow MIN$

Prioritization Example contd. (Basic Release Planning)



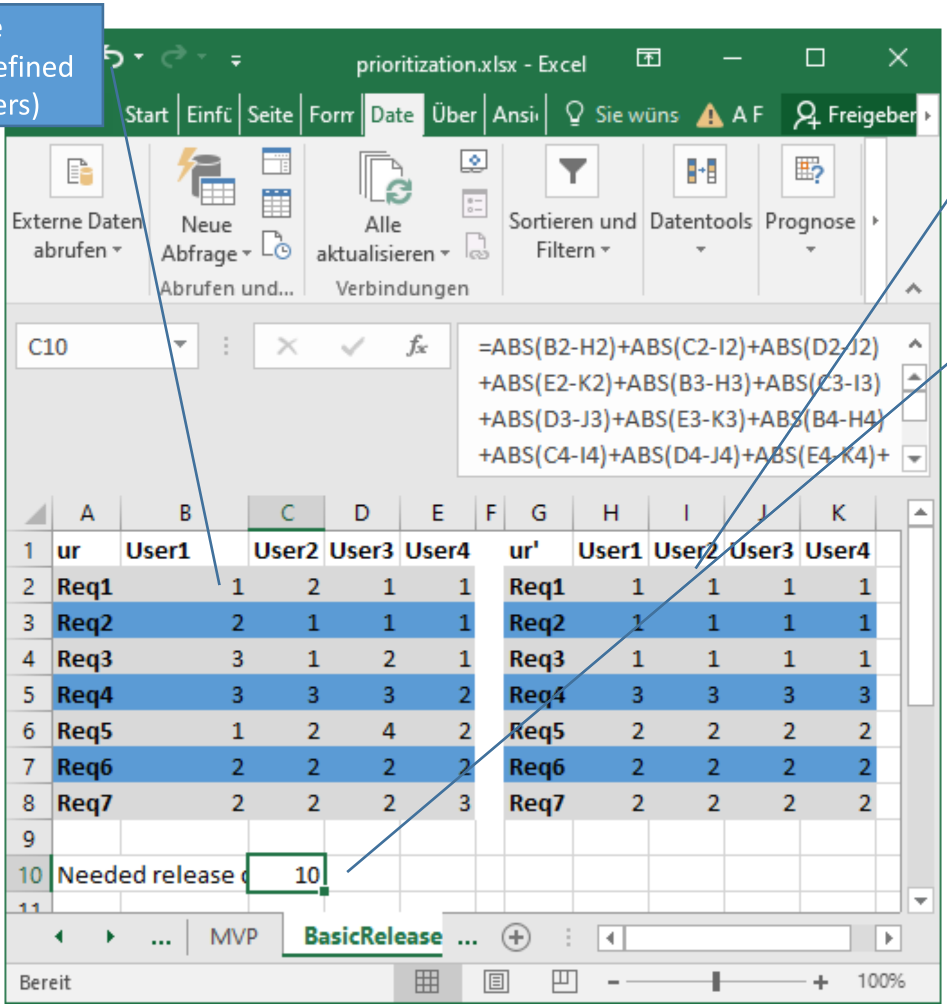
Requirement	User1	User2	User3	User4
Req1	1	2	1	1
Req2	2	1	1	1
Req3	3	1	2	1
Req4	3	3	3	2
Req5	1	2	4	2
Req6	2	2	2	2
Req7	2	2	2	3

- $ur_{11} = 1, ur_{21} = 2, ur_{31} = 1, .., ur_{47} = 3$
- $ur'_{11} = ur'_{21} = ur'_{31} = ur'_{41}$
- …
- $ur'_{17} = ur'_{27} = ur'_{37} = ur'_{47}$
- $ur'_{21}=1, ur'_{12}=1, ur'_{13}=1, ur'_{33}=1, ur'_{44}=3, ur'_{15}=2, ur'_{35}=2, ur'_{47}=2$
- For example: $chg_{21}=1, chg_{12}=1, \ldots$

- Change per user i: $chg_i = \sum_{k=1}^{n} chg_{ik}$
- Fairness aspect: $\sum_{i=1..m} \sum_{j=1..m} |chgi - chgj| \rightarrow MIN$

Prioritization Example contd. (MS Excel Solver)

preferred release assignments of Reqi defined by users (stakeholders)



C10

=ABS(B2-H2)+ABS(C2-I2)+ABS(D2-J2)+ABS(E2-K2)+ABS(B3-H3)+ABS(C3-I3)+ABS(D3-J3)+ABS(E3-K3)+ABS(B4-H4)+ABS(C4-I4)+ABS(D4-J4)+ABS(E4-K4)+

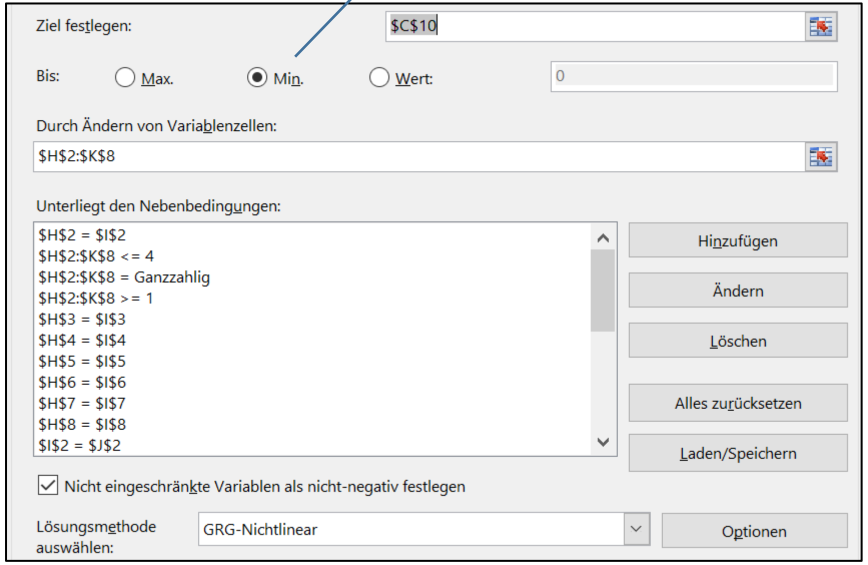
	A	B	C	D	E	F	G	H	I	J	K
1	ur	User1	User2	User3	User4		ur'	User1	User2	User3	User4
2	Req1	1	2	1	1		Req1	1	1	1	1
3	Req2	2	1	1	1		Req2	1	1	1	1
4	Req3	3	1	2	1		Req3	1	1	1	1
5	Req4	3	3	3	2		Req4	3	3	3	3
6	Req5	1	2	4	2		Req5	2	2	2	2
7	Req6	2	2	2	2		Req6	2	2	2	2
8	Req7	2	2	2	3		Req7	2	2	2	2
9											
10	Needed release c		10								

MVP | BasicRelease

minimal adaptations calculated by Excel Solver

optimization criteria: minimal release shifts compared to original user preferences

optimization criteria: minimize release shifts



Ziel festlegen: $C$10

Bis: ○ Max. ● Min. ○ Wert: 0

Durch Ändern von Variablenzellen:

$H$2:$K$8

Unterliegt den Nebenbedingungen:

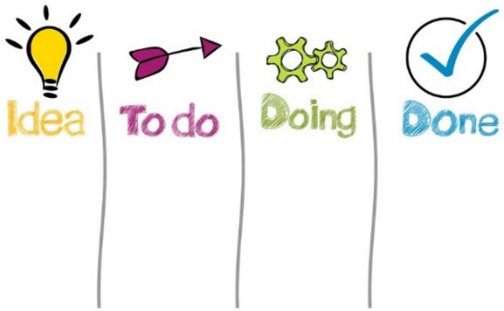
```
$H$2 = $I$2
$H$2:$K$8 <= 4
$H$2:$K$8 = Ganzzahlig
$H$2:$K$8 >= 1
$H$3 = $I$3
$H$4 = $I$4
$H$5 = $I$5
$H$6 = $I$6
$H$7 = $I$7
$H$8 = $I$8
$I$2 = $J$2
```

☑ Nicht eingeschränkte Variablen als nicht-negativ festlegen

Lösungsmethode auswählen: GRG-Nichtlinear

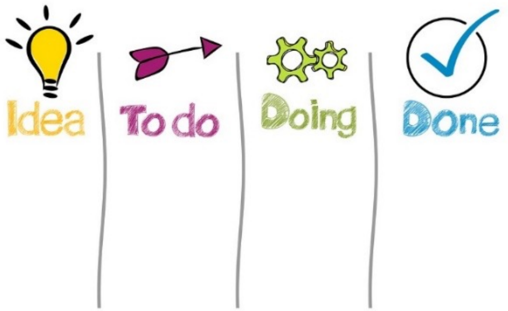
Remark: this .xlsx is provided in TeachCenter

Integrated Release Planning



- Goal: prioritization of the requirements to be implemented in the next *n releases* (special case: n=1, the *next-release problem*)
- Fine-granular requirements, similar to requirements specifications used in basic release planning
- Typically, dependency and resource constraints are taken into account (e.g., employee X is also engaged with 50% of his/her time in project Y)

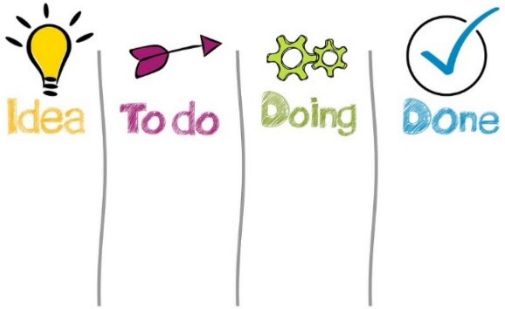
Integrated Release Planning: Constraints



Definition	Description
$req_i.rel = a$	req_i is assigned to release *a*
$req_i.rel < reqj.rel$	req_i must be implemented before req_j
$req_i.rel \leq reqj.rel$	req_j must not be implemented before req_i
$req_i.rel\ ! = reqj.rel$	req_i and req$_j$ must have different releases
$req_i.rel < a$	implementation of req_i before release *a*
$req_i.rel \geq a$	implementation of req_i not before release *a*
$req_i.rel = n\ or\ reqj.rel = n$	req_i or req_j not in release plan
$not(\lvert req_i.rel\ - reqj.rel\rvert > k)$	req_i and req_j must be implemented timely
$\lvert\{r\ in\ R: r.rel = rel\}\rvert \leq a$	not more than *a* requirements in release *rel*
$\sum_{req\ in\ R\ and\ req.rel=rel} req.dur \leq a$	not more than *a* hours bounded to *rel*

- req$_i$.rel = release of requirement i
- n = „empty“ release
- R = set of req$_i$
- req$_i$.dur = estimated implementation duration of req$_i$
- example task: find solution, balance implementation efforts

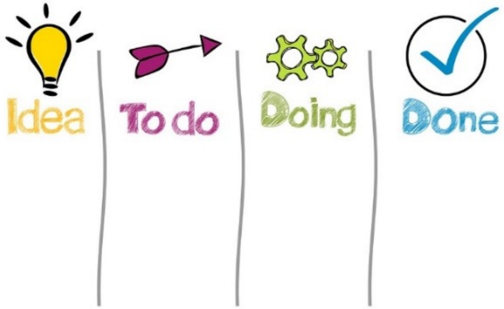
Prioritization Example contd. (Integrated Release Planning)



- Requirements R = $\{req_1, .., req_7\}$
- Release domains: 1..3 ($req_i.rel = x \rightarrow x \in \{1,2,3\}$)
- Implementation durations: $req_1.dur=2$, $req_2.dur=4$, $req_3.dur=3$, $req_4.dur=2$, $req_5.dur=4$, $req_6.dur=2$, $req_7.dur=1$
- Constraints: $req_1.rel <= req_2.rel$, $req_3.rel = 1$, $req_4.rel <= req_5.rel$, $req_6.rel = req_7.rel$
- Maximum implementation efforts per release: 8
- Optimization task: balance the implementation efforts

$$\sum_{i=1..\#releases} \sum_{j=1..\#releases} |efforts(i) - efforts(j)| \rightarrow MIN$$

Prioritization Example contd. (MS Excel Solver)

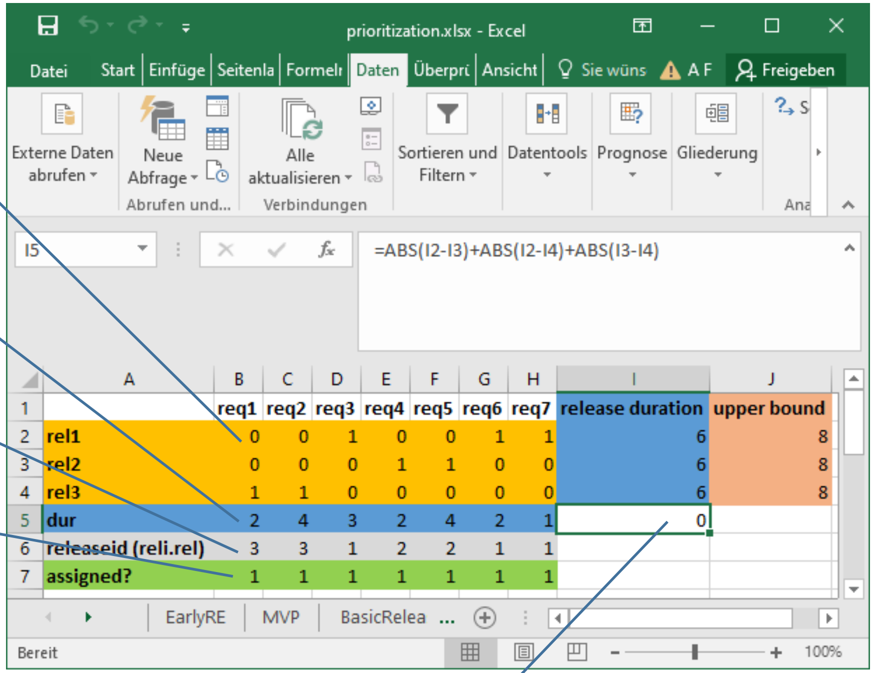


req$_i$ assigned to rel$_j$ (y/n)

estimated duration of a requirement

a requirement‘s release id

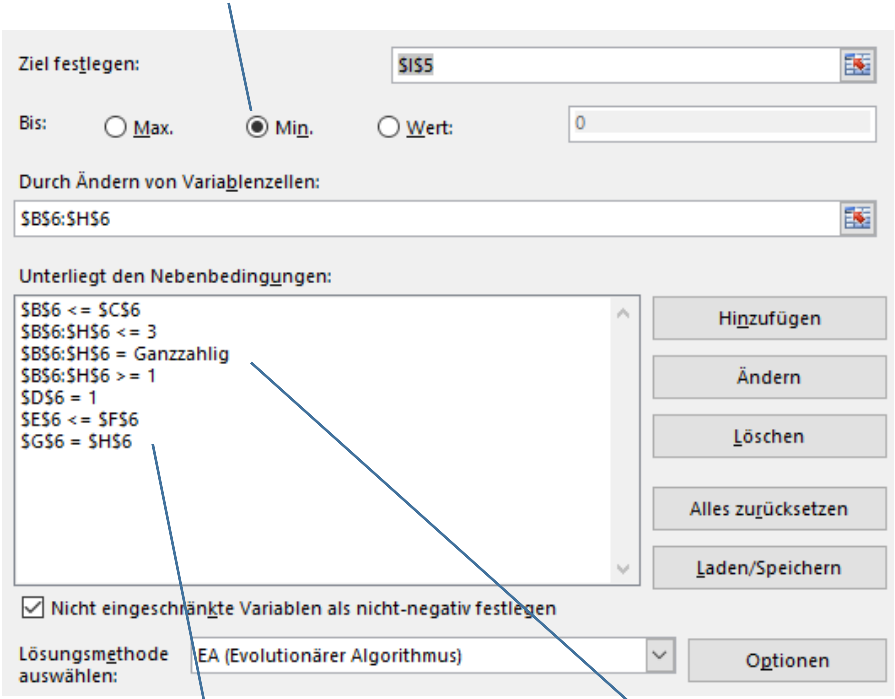
req$_i$ assigned to a release (y/n)



I5 =ABS(I2-I3)+ABS(I2-I4)+ABS(I3-I4)

	A	B	C	D	E	F	G	H	I	J
1		req1	req2	req3	req4	req5	req6	req7	release duration	upper bound
2	rel1	0	0	1	0	0	1	1	6	8
3	rel2	0	0	0	1	1	0	0	6	8
4	rel3	1	1	0	0	0	0	0	6	8
5	dur	2	4	3	2	4	2	1	0	
6	releaseid (reli.rel)	3	3	1	2	2	1	1		
7	assigned?	1	1	1	1	1	1	1		

optimization criteria: minimize difference in implementation efforts



Ziel festlegen: $I$5

Bis: ○ Max. ● Min. ○ Wert: 0

Durch Ändern von Variablenzellen:
$B$6:$H$6

Unterliegt den Nebenbedingungen:

```
$B$6 <= $C$6
$B$6:$H$6 <= 3
$B$6:$H$6 = Ganzzahlig
$B$6:$H$6 >= 1
$D$6 = 1
$E$6 <= $F$6
$G$6 = $H$6
```

☑ Nicht eingeschränkte Variablen als nicht-negativ festlegen

Lösungsmethode auswählen: EA (Evolutionärer Algorithmus)

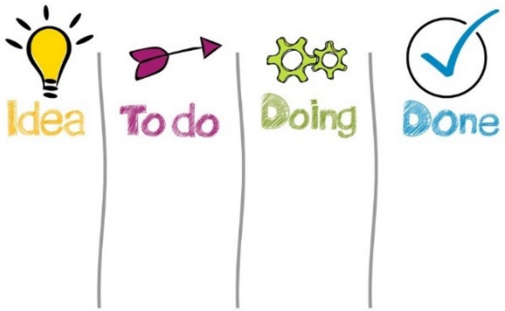
Remark: this .xlsx is provided in TeachCenter

optimization criteria/task (balancing the implementation effort)

constraints regarding release assignment

req$_i$ assigned to rel$_j$ (1..3)

Repetition (R3)

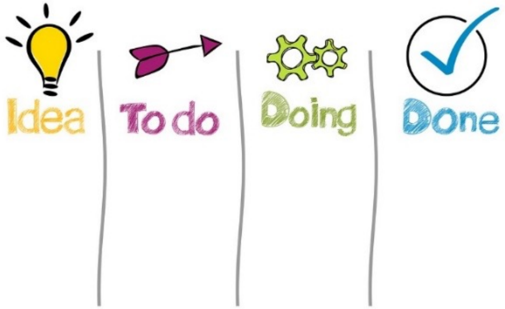


- Visit: https://checkr.tugraz.at/ (a TU Graz software).
- Login with your TU Graz student account (single sign-on supported).
- Enter the following participation code: **rJACCg** (note: you can try to answer the individual questions as often as you like!). No fixed time slots for the repetitions, **deadline for all repetitions**: **June 20th, 23:59:59**.
- Go to the category „Prioritization“ and answer the questions.
- Your answers will be taken into account as mentioned in the organization slides.

Thank You!

Univ.-Prof. DI Dr. Alexander Felfernig
Dr. Trang Tran
Applied Artificial Intelligence
Graz University of Technology, Austria

References



- **[RUH2010]** G. Ruhe. Product Release Planning – Methods, Tools, and Applications, CRC Press, 2010.
- **[QAQ2017]** R. Qaddoura, A. Abu-Srhan, M. Qasem, and A. Hudaib. Requirements Prioritization Techniques Review and Analysis, IEEE Conference on New Trends in Computing Sciences, pp. 258-263, 2017.