Appendix to the article: Integrating Informal Learning into Deployment Planning and Project Scheduling

APPENDIX A: Basic Data and Simulation Setting

Indices

Jobs (n = 16): A: Kick-Off; B: Specifications; C: Evaluation of Time-Requirements; D: System Structure; E: Project Calculation; F: Customer Coordination; G: 1. Frontend Programming; H: 1. Back-end Programming; I: Documentation; J: Team Meeting; K: Test Scenario creation; L: 2. Back-end Programming; M: 2. Frontend Programming; N: Team Meeting; O: Testing; P: Delivery
M = high, low; R = Product Owner, Scrum Master, Experienced Programmer, Unexperienced Programmer; T =N=135

Parameters

$a_{j,r,m}$

Job	PO		SM		EP		UP	
	high	low	high	low	high	low	high	low
A	1	1	1	1	5	5	3	3
B	1	1	1	1	2	1	2	1
C	1	1	1	1	2	1	2	2
D	0	0	0	0	2	1	2	1
E	1	1	1	0	0	0	0	0
F	1	1	1	0	0	0	0	0
G	0	0	0	0	3	2	2	1
H	0	0	0	0	3	2	2	1
I	1	1	1	0	1	0	0	0
J	1	1	1	1	5	5	3	3
K	0	0	1	0	2	0	2	1
L	0	0	0	0	3	2	2	1
M	0	0	0	0	3	2	2	1
N	1	1	1	1	5	5	3	3
O	1	1	0	0	1	0	2	1
P	1	1	1	1	2	2	0	0

$\lambda_{j,r,m}$

Jobs

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
PO.high	1	1	1	0	1	1	0	0	1	1	0	0	0	1	1	1
SM.high	1	1	1	0	1	1	0	0	1	1	1	0	0	1	0	1
EP.high	1	1	1	1	0	0	1	1	1	1	1	1	1	1	1	1
UP.high	1	1	1	1	0	0	1	1	0	1	1	1	1	1	1	0
PO.low	1	1	1	0	1	1	0	0	1	1	0	0	0	1	1	1
SM.low	1	1	1	0	0	0	0	0	0	1	0	0	0	1	0	1
EP.low	1	1	1	1	0	0	1	1	0	1	0	1	1	1	0	1
UP.low	1	1	1	1	0	0	1	1	0	1	0	1	1	1	1	0

$p_{j,m}$

Jobs

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
High	1	1	1	1	1	5	30	15	10	2	5	20	30	2	10	1
Low	1	2	2	2	2	7	40	20	15	2	8	25	40	2	15	1

$K_{r,t}$: for all t: PO=1; SM=1; EP=5; UP=3

Critical path data

Jobs

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
EST_j	1	2	3	3	4	5	10	4	40	40	42	42	42	72	74	84
LET_j	52	53	54	75	55	60	90	90	124	92	124	122	122	124	134	135

(+50 additional security buffer for optimization)

Simulation setting

$w = [0.1,1.0]$; $w = \frac{1}{2} \cdot \alpha$; $\alpha = \sum_{j=1}^{A} a_j$

Table A0: Simulation scenarios for analysis

Scenario	Label	Activities with informal learning
0	Baseline scenario (project scheduling without informal learning)	-
1	Total value-adding activities (continuous informal learning)	G, H, L, M
2	First Level Programming	G,H
3	Second Level Programming	L,M
4	Mixed-Learning 1: Frontend-Backend	G, L
5	Mixed-Learning 2: Backend-Frontend	H, M

APPENDIX B: Baseline Scenario 0, Unsystematic Informal Learning

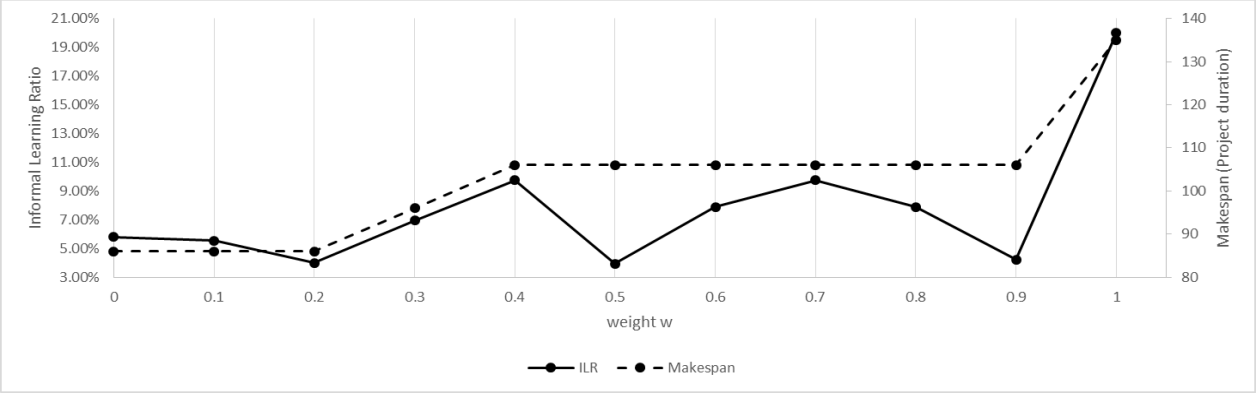
$\boldsymbol{w = [0; 1.0]; \alpha = 0}$

Table A1: Simulations for the baseline scenario

Scenario 0: Baseline	Quality (accumulated team days)	Project duration (makespan)	ILR %	optimal solution computation time (seconds) Time limit 3600 seconds
Critical Path Method	310	84	-	-
Alternative 0 (w=0)	310	86	5.81	90
Alternative 1 (w=0.1)	325	86	5.54	140
Alternative 2 (w=0.2)	325	86	4.00	881
Alternative 3 (w=0.3)	360	96	6.94	579
Alternative 4 (w=0.4)	380	106	9.74	1023
Alternative 5 (w=0.5)	380	106	3.95	610
Alternative 6 (w=0.6)	380	106	7.89	220
Alternative 7 (w=0.7)	380	106	9.74	223
Alternative 8 (w=0.8)	380	106	7.89	170
Alternative 9 (w=0.9)	380	106	4.21	151
Alternative 10 (w=1)	380	135	20.00	41

The baseline scenario includes project scheduling without explicit informal learning factors, that is, unsystematic informal learning and all α=0. The scenario hence reflects regular quantitative project scheduling with multiple objectives focusing efficiency (Hartmann and Briskorn, 2010). The total value-adding activities from Scenario 1 (first and second level programming) are taken into account for ILR calculation. The scenario does not consider explicit informal learning. However, based on the model structure including activity-splitting and preemption, there is a minor potential for informal learning behaviors. The makespan (project duration) is expectedly at the lowest level and in most of the cases stable, only a complete focus on quality (w=1) requires a full project horizon of N=135 days.

Figure A1: ILR and Makespan development baseline scenario



APPENDIX C: Scenario 1, total value-adding activities (G, H, L, M)

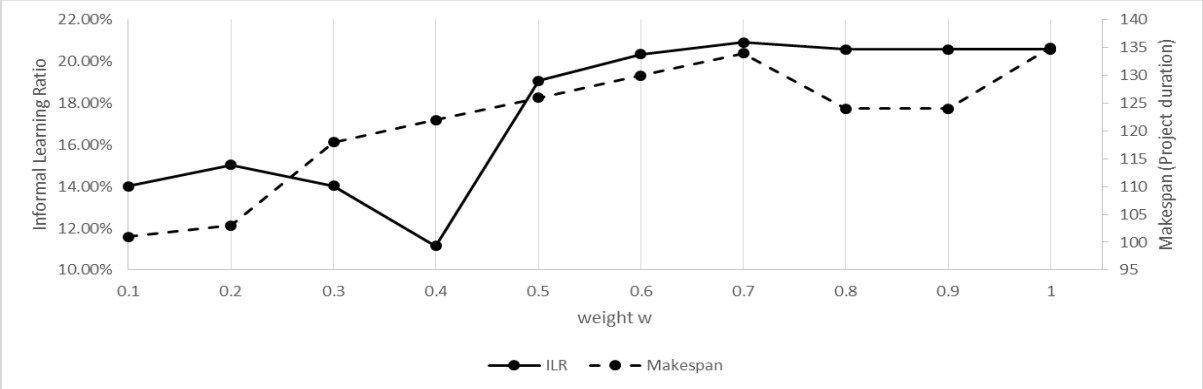
$$\boldsymbol{w = [0.1; 1.0]; \alpha > 0}$$

Table A2: Simulations for learning in activities G, H, L, M

Scenario 1: Value-adding activities	Quality (accumulated team days)	Project duration (makespan)	α-Split equally for G,H,L,M	ILR %	optimal solution computation time (seconds) Time limit 3600 seconds
Alternative 1 (w=0.1)	364	101	0.05	14.01	464
Alternative 2 (w=0.2)	372	103	0.10	15.05	1778
Alternative 3 (w=0.3)	406	118	0.15	14.04	1580
Alternative 4 (w=0.4)	430	122	0.20	11.16	1071
Alternative 5 (w=0.5)	430	126	0.25	19.07	416
Alternative 6 (w=0.6)	442	130	0.30	20.36	383
Alternative 7 (w=0.7)	454	134	0.35	20.93	229
Alternative 8 (w=0.8)	452	124	0.40	20.58	293
Alternative 9 (w=0.9)	452	124	0.45	20.58	172
Alternative 10 (w=1)	452	135	0.50	20.58	85

Scenario 1 includes all value-adding activities in the considered IT project, i.e. first and second level programming (G, H, L, M). In this sense, this 'extreme' scenario reflects systematic learning in the project. Thus, the scenario shows the highest informal learning ratios in the analysis with a maximum at w=0.7 and a smoothing effect for project durations already at lower weights (w >0.3). However, the durations are not higher compared to the split scenarios 2 to 5.

Figure A2: ILR and Makespan development scenario 1



APPENDIX D: Scenario 2, first-level programming (G, H)

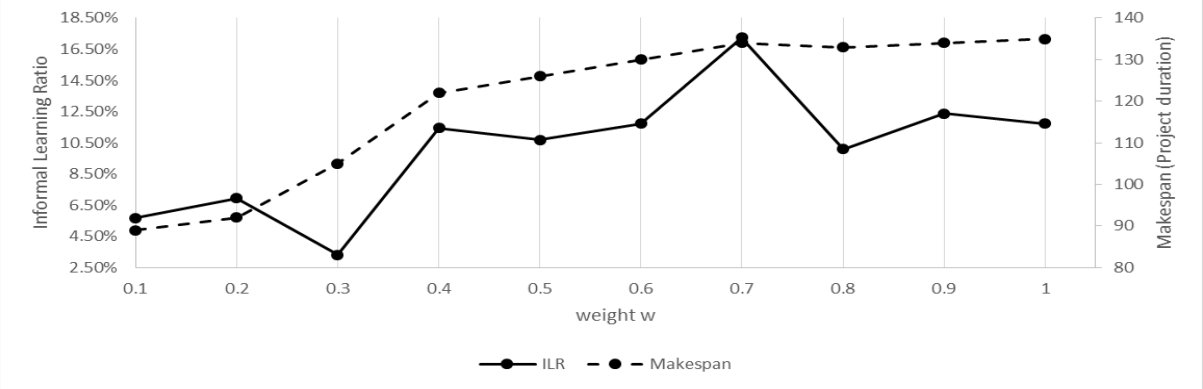
$w = [0.1; 1.0]; \alpha > 0$

Table A3: Simulations for learning in activities G,H

Scenario 2: First Level Programming	Quality (accumulated team days)	Project duration (makespan)	α-Split equally for G,H	ILR %	optimal solution computation time (seconds) Time limit 3600 seconds
Alternative 1 (w=0.1)	335	89	0.1	5.67	385
Alternative 2 (w=0.2)	345	92	0.2	6.96	728
Alternative 3 (w=0.3)	390	105	0.3	3.33	531
Alternative 4 (w=0.4)	428	122	0.4	11.45	462
Alternative 5 (w=0.5)	440	126	0.5	10.68	200
Alternative 6 (w=0.6)	452	130	0.6	11.73	186
Alternative 7 (w=0.7)	464	134	0.7	17.24	124
Alternative 8 (w=0.8)	456	133	0.8	10.09	152
Alternative 9 (w=0.9)	461	134	0.9	12.36	120
Alternative 10 (w=1)	460	135	1	11.74	49

Scenario 2 considers the first-level programming (G, H). Here, systematic informal learning is limited to the beginning of the project. There is an increase in makespan with decreasing intensity. Both ILR and project duration reach a peak at w=0.7. In general, this scenario shows weak performance with high project durations and thus low efficiency and high ILR volatility.

Figure A3: ILR and Makespan development scenario 2



APPENDIX E: Scenario 3, second level programming (L, M)

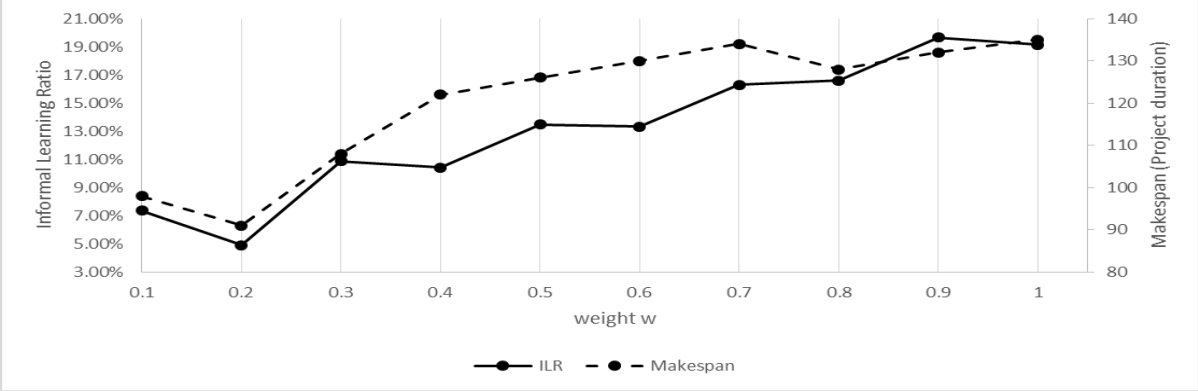
$w = [0.1; 1.0]; \alpha > 0$

Table A4: Simulations for learning in activities L,M

Scenario 3: Second Level Programming	Quality (accumulated team days)	Project duration (makespan)	α-Split equally for L,M	ILR %	optimal solution computation time (seconds) Time limit 3600 seconds
Alternative 1 (w=0.1)	354	98	0.1	7.34	464
Alternative 2 (w=0.2)	347	91	0.2	4.90	241
Alternative 3 (w=0.3)	386	108	0.3	10.88	1071
Alternative 4 (w=0.4)	432	122	0.4	10.42	435
Alternative 5 (w=0.5)	430	126	0.5	13.49	1000
Alternative 6 (w=0.6)	458	130	0.6	13.32	199
Alternative 7 (w=0.7)	454	134	0.7	16.30	179
Alternative 8 (w=0.8)	464	128	0.8	16.59	178
Alternative 9 (w=0.9)	458	132	0.9	19.65	205
Alternative 10 (w=1)	486	135	1	19.14	174

Scenario 3 considers the second level programming (L, M). Here, systematic informal learning is limited to the end of the project. Here, the informal learning ratios are constantly increasing. In this respect, using informal learning is more effective in later project phases compared to scenario 2. However, also project durations are on a constant high level.

Figure A4: ILR and Makespan development scenario 3



APPENDIX F: Scenario 4, mixed learning 1 (G, L)

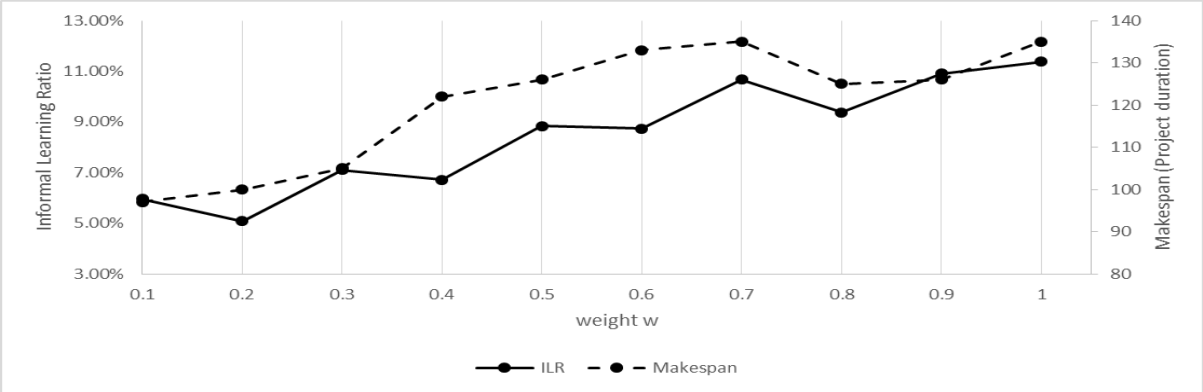
$w = [0.1; 1.0]; \alpha > 0$

Table A5: Simulations for learning in activities G, L

Scenario 4: Mixed Learning 1:	Quality (accumulated team days)	Project duration (makespan)	α-Split equally for G, L	ILR %	optimal solution computation time (seconds) Time limit 3600 seconds
Alternative 1 (w=0.1)	352	97	0.1	5.97	262
Alternative 2 (w=0.2)	374	100	0.2	5.08	3600
Alternative 3 (w=0.3)	380	105	0.3	7.11	1.360
Alternative 4 (w=0.4)	432	122	0.4	6.71	795
Alternative 5 (w=0.5)	430	126	0.5	8.84	722
Alternative 6 (w=0.6)	458	133	0.6	8.73	430
Alternative 7 (w=0.7)	450	135	0.7	10.67	3600
Alternative 8 (w=0.8)	448	125	0.8	9.38	359
Alternative 9 (w=0.9)	440	126	0.9	10.91	446
Alternative 10 (w=1)	466	135	1	11.37	136

Scenario 4 partially combines first and second level programming (G, L). Learning is split to different activities in different project stages. Similar to scenario 3 there is a constant increase in informal learning ratios, but on significant lower level. In this respect, learning in earlier stages shows an impairing impact on informal learning and project performance.

Figure A5: ILR and Makespan development scenario 4



APPENDIX G: Scenario 5, mixed learning 2 (H, M)

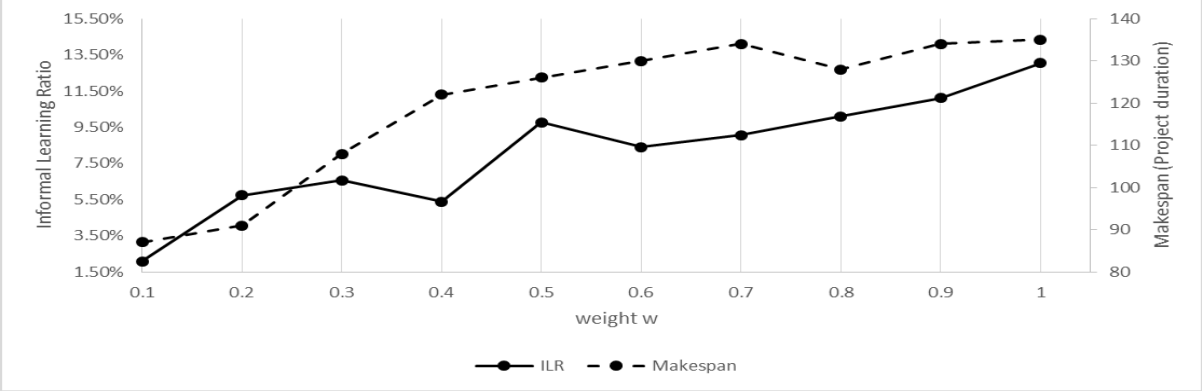
$w = [0.1; 1.0]; \alpha > 0$

Table A6: Simulations for learning in activities H, M

Scenario 5: Mixed Learning 2:	Quality (accumulated team days)	Project duration (makespan)	α-Split equally for H, M	ILR %	optimal solution computation time (seconds) Time limit 3600 seconds
Alternative 1 (w=0.1)	335	87	0.1	2.09	726
Alternative 2 (w=0.2)	349	91	0.2	5.73	1191
Alternative 3 (w=0.3)	396	108	0.3	6.57	1521
Alternative 4 (w=0.4)	428	122	0.4	5.37	989
Alternative 5 (w=0.5)	440	126	0.5	9.77	840
Alternative 6 (w=0.6)	452	130	0.6	8.41	404
Alternative 7 (w=0.7)	464	134	0.7	9.05	210
Alternative 8 (w=0.8)	456	128	0.8	10.09	271
Alternative 9 (w=0.9)	468	134	0.9	11.11	159
Alternative 10 (w=1)	460	135	1	13.04	143

AS in the scenario 4, the scenario 5 partially combines first and second level programming (H, M). Learning is split to different activities in different project stages. Here, there is a reinforcing increase in informal learning ratios for w≥0.6. Again, learning in earlier stages shows an impairing impact on informal learning and project performance.

Figure A6: ILR and Makespan development scenario 5



APPENDIX H: GAMS source code

```
***MRCPS SLE***
set
*Jobs
J //
*Resources
R //
M //
*Time periods
T //;
*symbols in job index with j as successor
alias (i,j);
*Variables
variables
MAKESPAN
GOAL
Q;
positive variable
y(j,m)
x(j,r,m,t);
positive variables
s(j)
c(j);
*AoN
set
path(i,j) //;
parameters
*Earliest starting time
FAZ(j) //
*Latest finishing time +50 days computation buffer
SEZ(j)//
N //
w //;
table alpha(j,m);
table p(j,m);
table a(j,r,m);
table lambda(j,r,m);
table Renewable(t,r);
equations;
objective..          GOAL =E= ((w)*Q)-((1-w)*c('Del'));
NB1(j)..             sum((m), y(j,m)) =E=1;
NB2(j,r,m)..         sum((t), x(j,r,m,t)) =E= (p(j,m)*(1+alpha(j,m))*y(j,m))*lambda(j,r,m);
NB3(i,j)$path(i,j).. c(i) =L= s(j)-1;
NB4(j,r,m,t)..       s(j)=L= x(j,r,m,t)*ord(t)+N*(1-x(j,r,m,t));
NB5(j,r,m,t)..       c(j)=G= x(j,r,m,t)*ord(t);
NB6(j)..             s(j) =G= FAZ(j);
NB7(j)..             c(j) =L= SEZ(j);
NB8(r,t)..           sum((j,m), a(j,r,m)*x(j,r,m,t)) =L= Renewable(t,r);
NB9..                sum((j,r,m),(p(j,m)*(1+alpha(j,m))*y(j,m))*lambda(j,r,m)) =E= Q;
NB10..               MAKESPAN=E= c('Del');
model Modell /all/ ;
option mip=scip;
option optcr=0;
option resLim=3600;
solve Modell using mip maximizing GOAL;
```