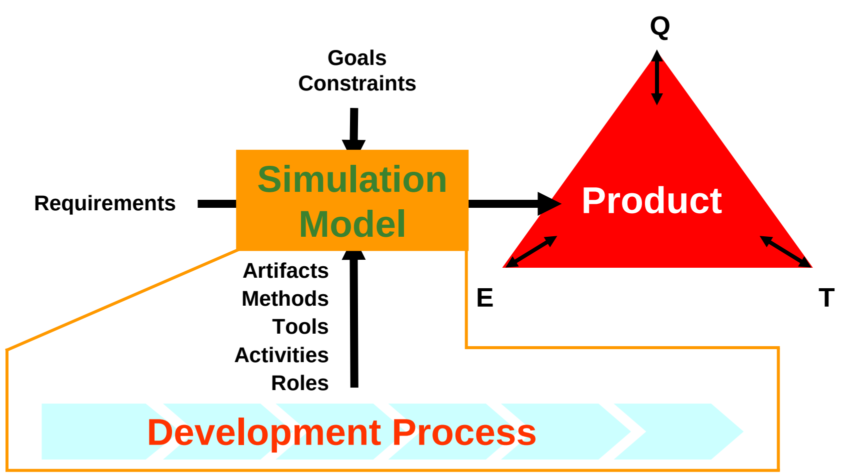
[simula . research laboratory]

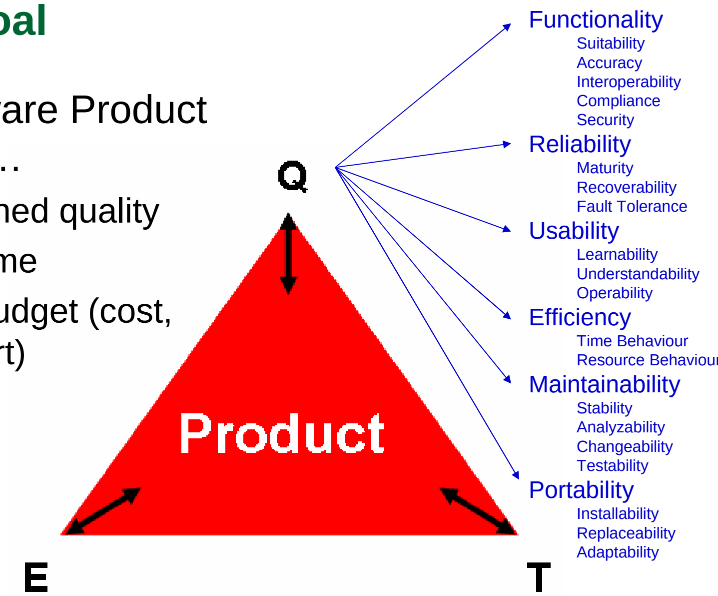
Meeting the Quality Goals – Better Software Products through Accelerated Technology Evaluation in a Virtual Software Production Laboratory (VSPL)

What is Software Process Simulation?



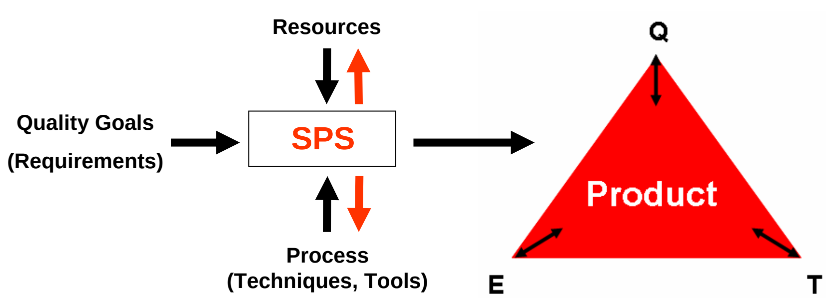
The Goal

- Software Product with …
  - defined quality
  - in time
  - in budget (cost, effort)



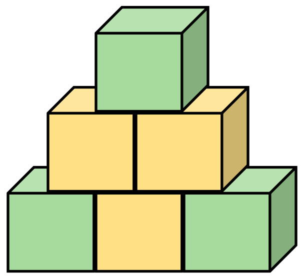
The Issue – SPS Application

- What V&V Techniques to apply …
- when? / by whom? / at which intensity level?
- … to achieve the product quality goals under given effort and time constraints



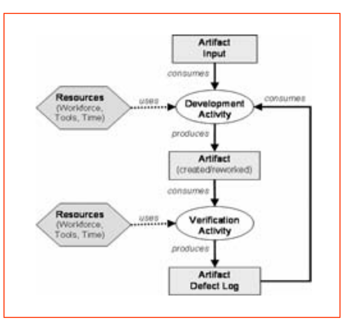
The Issue – SPS Modeling

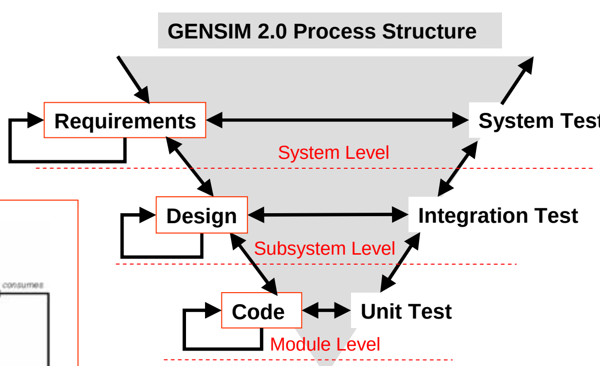
- Process Simulation Modeling is costly
  - Complex
  - Each time done from scratch
- Have an agile modeling process (i.e., Agile-IMMoS) and a set of customizable and reusable model building blocks.



Simulation Model

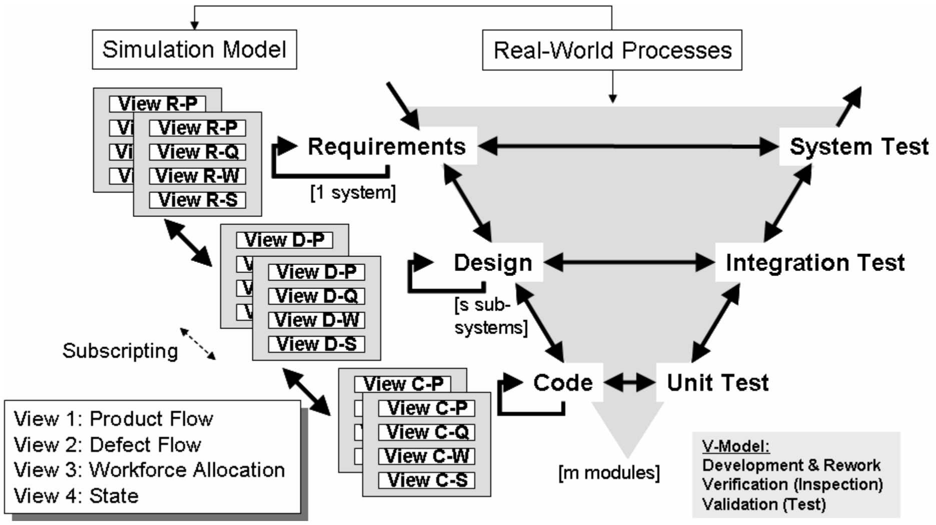
SPS Model – Process Architecture Example





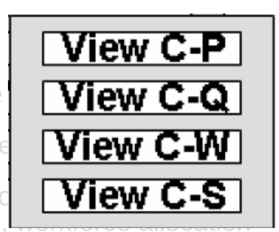
V-Model:
Development,
Verification (Inspection),
Validation (Test)

SPS Model – Process Architecture Example



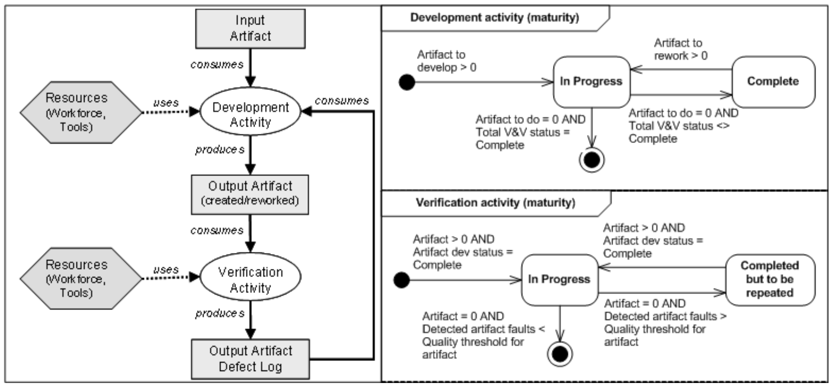
Prototype Implementation: GENSIM 2.0

- GENSIM 2.0 is implemented in Vensim®
- Besides applying macro-patterns, 3 features of Vensim® were used to add to the reusability of GENSIM 2.0:
  - **Views**: to capture the main dimensions of project performance, i.e., duration, cost and quality as well as the states of the software development process
    - Increased Understandability
  - **Subscripting**: to model individual software
    - Customizable to different projects with differe
  - **Dynamic Link Libraries (DLL)**: to extract c policies and heuristics from the model, e.g., workforce allocation
    - Increased adaptability to various organizations



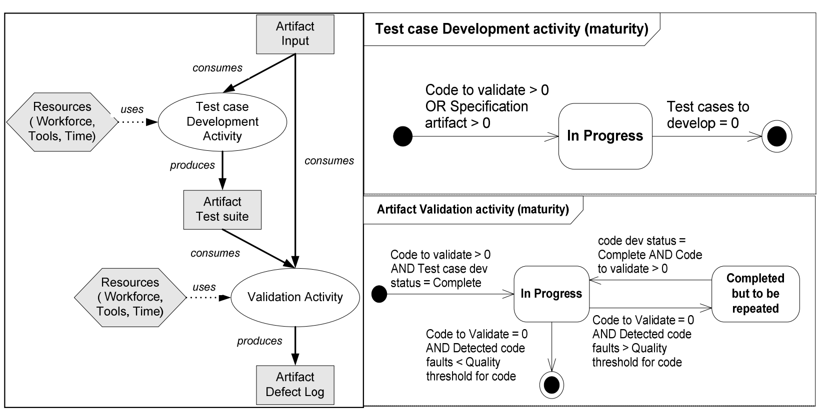
SPS Model – Macro-Patterns

- Process and State Views – Development & Verification



SPS Model – Macro-Patterns

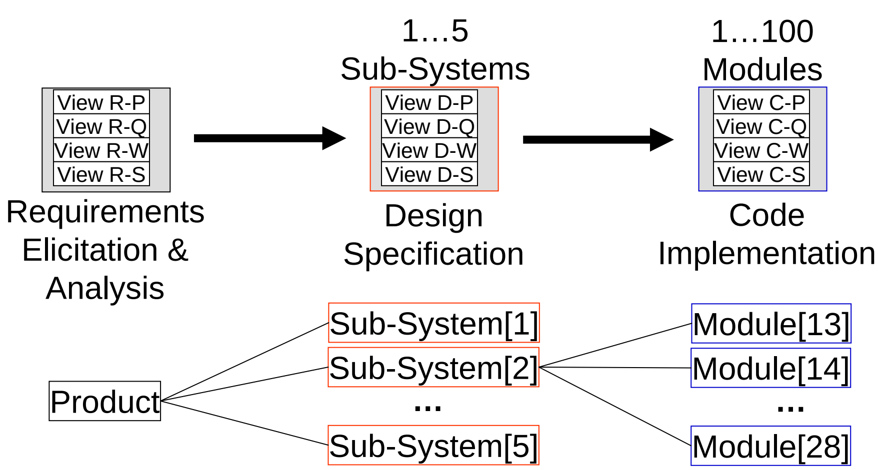
- Process and State Views – Test Case Development & Validation



Prototype Implementation: GENSIM 2.0

- GENSIM 2.0 is implemented in Vensim®
- Besides applying macro-patterns, 3 features of Vensim® were used to add to the reusability of GENSIM 2.0:
  - **Views**: to capture the main dimensions of project performance, i.e., duration, cost and quality as well as the states of the software development process
    - Increased Understandability
  - **Subscripting**: to model individual software artifacts
    - Customizable to different projects with different (types of) products
  - **Dynamic Link Libraries (DLL)**: to extract organization-specific policies and heuristics from the model, e.g., workforce allocation
    - Increased adaptability to various organizations

SPS Model – Product Structure



Prototype Implementation: GENSIM 2.0

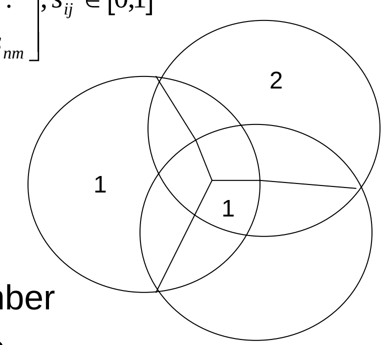
- GENSIM 2.0 is implemented in Vensim®
- Besides applying macro-patterns, 3 features of Vensim® were used to add to the reusability of GENSIM 2.0:
  - **Views**: to capture the main dimensions of project performance, i.e., duration, cost and quality as well as the states of the software development process
    - Increased Understandability
  - **Subscripting**: to model individual software artifacts
    - Customizable to different projects with different (types of) products
  - **Dynamic Link Libraries (DLL)**: to extract organization-specific policies and heuristics from the model, e.g., workforce allocation
    - Increased adaptability to various organizations

SPS Model – External DLLs

- Heuristic for Developer Allocation to Tasks
- Skill Matrix

$$S_{n\times m} = \begin{bmatrix} s_{11} & \cdots & s_{1m} \\ \vdots & \ddots & \vdots \\ s_{n1} & \cdots & s_{nm} \end{bmatrix}, s_{ij} \in [0,1]$$

- Assignment Algorithm
  - E.g.: Assign available developers to waiting activities (tasks) depending on task weight and the number of tasks a developer can do.



SPS Model – Parameters

- Model parameters related to ***code development and verification***

  (list not complete)

	Parameter Name	Attribute	Type	View
1	Verify code or not	Process	Input	C-P
2	# of modules per subsystem	Product	Input	C-P
3	Code doc quality threshold per size unit	Project	Input	C-Q
4	Required skill level for code dev	Project	Input	C-W
5	Required skill level for code ver	Project	Input	C-W
6	Developers' capabilities for code dev	People	Input	C-W
7	Developers' capabilities for code ver	People	Input	C-W
8	Maximum code ver. effectiveness	Process	Calibrated	C-P
9	Maximum code ver. rate per person per day	Process	Calibrated	C-P
10	Average design to code conversion factor	Product	Calibrated	C-P
11	Average # of UT test cases per code size unit	Product	Calibrated	C-P
12	Minimum code fault injection rate per size unit	Product	Calibrated	C-Q
13	Average design to code fault multiplier	Product	Calibrated	C-Q
14	Code rework effort for code faults detected in CI	Product	Calibrated	C-Q
15	Code rework effort for code faults detected in UT	Product	Calibrated	C-Q
16	Code rework effort for code faults detected in IT	Product	Calibrated	C-Q
17	Code rework effort for code faults detected in ST	Product	Calibrated	C-Q
18	Initial code dev. rate per person per day	People	Calibrated	C-W
19	Initial code ver. rate per person per day	People	Calibrated	C-W
20	Code doc size (actual)	Product	Output	C-P
21	Code to rework (actual)	Process	Output	C-P
22	Code development rate (actual)	Process	Output	C-P
23	Code verification rate (actual)	Process	Output	C-P
24	Code faults undetected	Product	Output	C-Q
25	Code faults detected	Product	Output	C-Q
26	Code faults corrected	Product	Output	C-Q
27	Code dev. effort (incl. rework; actual)	Process	Output	C-W
28	Code ver. effort (actual)	Process	Output	C-W

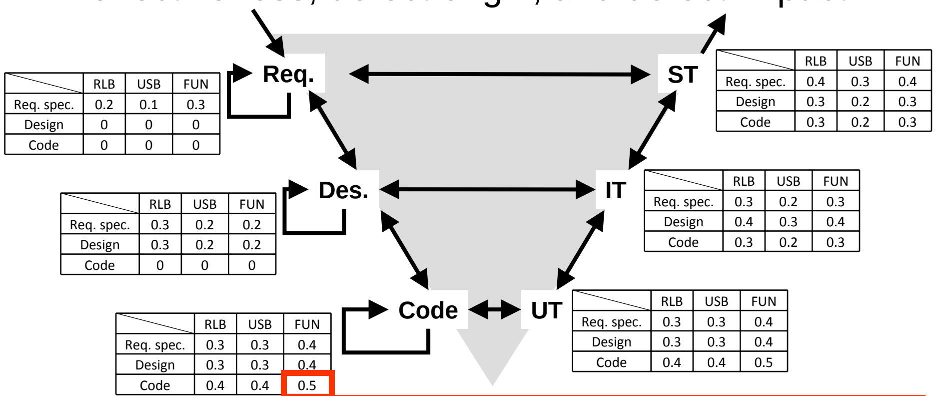
#=number, CI=Code Inspection, UT=Unit Test, IT=Integration Test, ST=System Test

SPS Model – Calibration

- Different sources for calibration:
  - Expert opinion
  - Organization-Specific repositories
  - Public repositories (often cross-organizational)
  - SE Literature
- Detailed specification of all parameters and the way they can be calibrated allows for calibrating GENSIM 2.0 to any of the above sources.

SPS Model – Calibration Example 1

- Assumptions (or data) about defect detection effectiveness, defect origin, and defect impact:



Req.:

	RLB	USB	FUN
Req. spec.	0.2	0.1	0.3
Design	0	0	0
Code	0	0	0

Des.:

	RLB	USB	FUN
Req. spec.	0.3	0.2	0.2
Design	0.3	0.2	0.2
Code	0	0	0

Code:

	RLB	USB	FUN
Req. spec.	0.3	0.3	0.4
Design	0.3	0.3	0.4
Code	0.4	0.4	0.5

UT:

	RLB	USB	FUN
Req. spec.	0.3	0.3	0.4
Design	0.3	0.3	0.4
Code	0.4	0.4	0.5

IT:

	RLB	USB	FUN
Req. spec.	0.3	0.2	0.3
Design	0.4	0.3	0.4
Code	0.3	0.2	0.3

ST:

	RLB	USB	FUN
Req. spec.	0.4	0.3	0.4
Design	0.3	0.2	0.3
Code	0.3	0.2	0.3

Code inspections find 50% of all defects in the code that affect the quality characteristic “Functionality” and that were injected due to a programming error.

SPS Model – Calibration Example 2

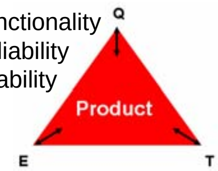
- Reported data on defect injection, detection, and rework effort
- Sources:
  - [5] Damm L, Lundberg L, Wohlin C (2006) Faults-slip-through - a concept for measuring the efficiency of the test process. Software Process: Improvement and Practice 11(1): 47-59
  - [8] Frost A, Campo M (2007) Advancing Defect Containment to Quantitative Defect Management. CrossTalk – The Journal of Defense Software Engineering 12(20): 24-28
  - [24] Wagner S (2006) A Literature Survey of the Quality Economics of Defect-Detection Techniques. ISESE 2006, pp 194-203

Calibration Parameter	Value	
	Calibration A	Calibration B
Minimum code fault injection rate per size unit	14.5 Defect/KLOC [8]	
Maximum code verification effectiveness	0.53 [8]	
Max. code verification rate per person per day	0.6 KLOC/PD [24]	
Code rework effort for code faults detected in CI	0.34 PD/Def. [24]	
Code rework effort for code faults detected in UT	0.43 PD/Def. [24]	
Code rework effort for code faults detected in IT	0.68 PD/Def. [24]	1.08 PD/Def. [5, 24]
Code rework effort for code faults detected in ST	1.05 PD/Def. [24]	5.62 PD/Def. [5, 24]

KLOC: Kilo Lines of Code, PD: Person-Day, Def: Defect.

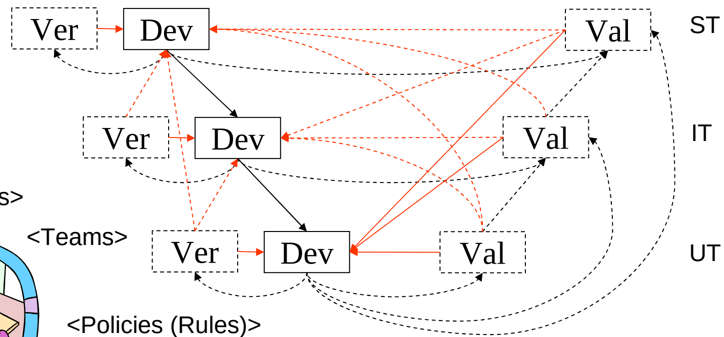
SPS Model – Application

Functionality
Reliability
Usability



- Current Model

Requirements Analysis and Specification (System)
Design (Sub-Systems)
Implementation (Modules)



<Individuals>

<Teams>

<Policies (Rules)>

SPS Model Application Example

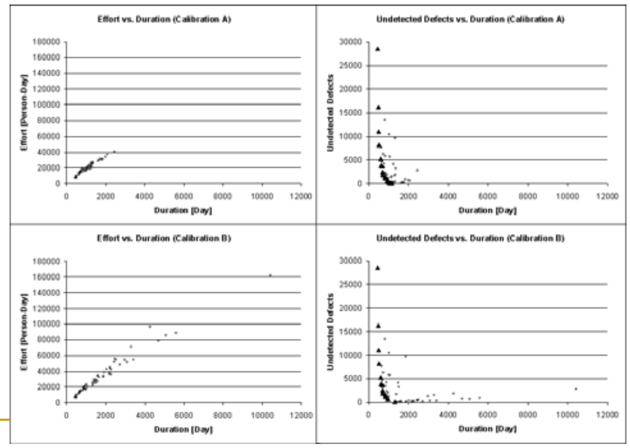
- Scenario 1:
  Impact of different combinations of verification and validation (V&V) activities on project duration, product quality, and effort.
  - Verification activities include Requirements Inspections (RI), Design Inspections (DI) and Code Inspections (CI).
  - Validation activities include Unit Test (UT), Integration Test (IT), and System Test (ST).
  - Per V&V activity exactly one technique with given efficiency and effectiveness is available.
  - A V&V technique is either applied to all documents of the related type (e.g., requirements, design, and code documents) or it is not applied at all.

SPS Model Application Example

- Scenario 1:
  Impact of different combinations of verification and validation (V&V) activities on project duration, product quality, and effort.

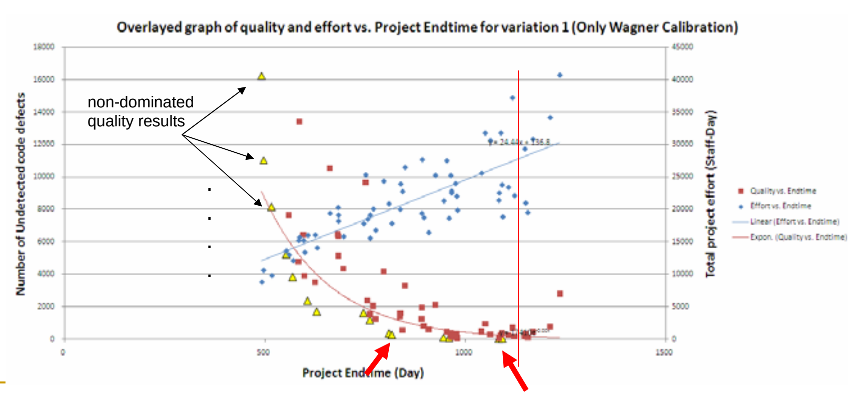
Calibration A

Calibration B



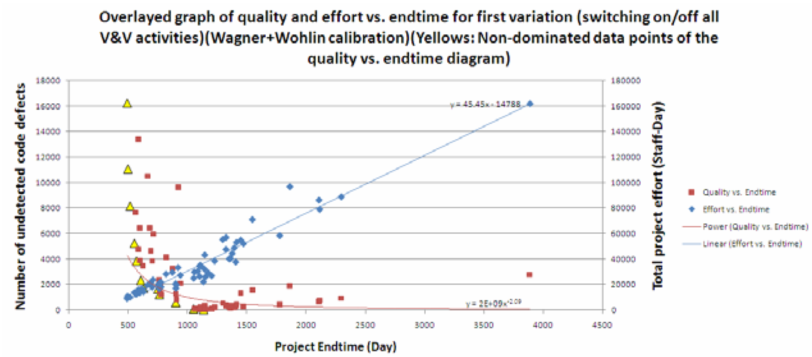
SPS Model Application Example

- Scenario 1:
  Impact of different combinations of verification and validation (V&V) activities on project duration, product quality, and effort.
  - Results with Calibration A:



SPS Model Application Example

- Scenario 1:
  Impact of different combinations of verification and validation (V&V) activities on project duration, product quality, and effort.
  - Results with Calibration B:



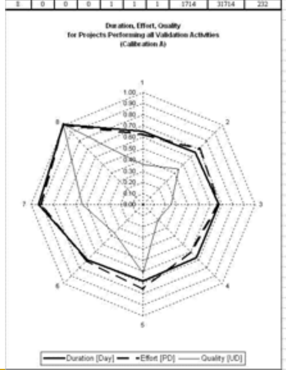
SPS Model Application Example

- Scenario 1:
  Impact of different combinations of verification and validation (V&V) activities on project duration, product quality, and effort.
  - Comparison of results for those cases where all test activities are performed.

    ➔ Performing <u>all verification</u> activities is optimal only for Calibration B.

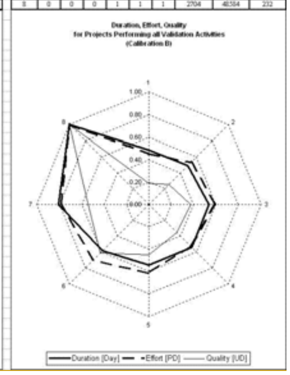
Calibration A

Case	RI	DI	CI	UT	IT	ST	Calibration A: Duration [Day]	Effort [PD]	Quality [UD]
1	1	1	0	1	1	1	1114	19666	83
2	0	1	0	1	1	1	1117	22374	104
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
4	1	1	1	1	1	1	1134	18924	44
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
6	1	0	1	1	1	1	1198	22483	84
7	0	0	1	1	1	1	1390	20979	126
8	0	0	0	1	1	1	1714	31714	232



Calibration B

							Calibration B: Duration		Quality
1	1	1	1	1	1	1	1308	22160	44
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
3	1	0	1	1	1	1	1449	28818	86
4	1	1	0	1	1	1	1463	25545	83
5	0	1	0	1	1	1	1468	29790	104
6	1	0	0	1	1	1	1571	34080	142
7	0	0	1	1	1	1	2177	38223	126
8	0	0	0	1	1	1	2704	48384	232



SPS Model Application Example

- Scenario 2:
  - Impact of different combinations of verification activities on project duration, product quality, and effort.
  - Alternative Inspection Techniques
    - T-type 10% more effective but 25% less efficient than S-Type
  - All test activities are conducted

		Requirements Inspection (RI)	Design Inspection (DI)	Code Inspection (CI)
S-type	Effective-ness	75%	76%	53%
	Efficiency	8 Pages/PD	30 Pages/PD	0.6 KLOC/PD
T-type	Effective-ness	82.5%	83.6%	58.3%
	Efficiency	6 Pages /PD	22.5 Pages /PD	0.45 KLOC /PD

KLOC: Kilo Lines of Code, PD: Person-Day

SPS Model Application Example

- Scenario 2 – Simulation Results:

	Row	RI	DI	CI	RI-tech.	DI-tech.	CI-tech.	Duration [Day]	Effort [PD]	Quality [UD]
D	1	0	1	1		T	S	*1281	24163	49
D	2	1	1	1	S	T	S	1296	21341	39
D	3	1	1	1	T	T	S	1297	20996	37
	4	0	1	1		T	T	1299	24068	47
D	5	1	1	1	S	T	T	1302	21189	36
D	6	1	1	1	T	T	T	1306	*20881	*35
	7	1	1	1	S	S	S	1308	22160	44
	8	1	1	1	T	S	T	1313	21610	39
	9	1	1	1	T	S	S	1313	21769	42
	10	1	1	1	S	S	T	1323	21989	41
	11	0	1	1		S	S	1326	25697	58
	12	0	1	1		S	T	1333	25395	54
	13	1	0	1	T		S	1417	27409	77
	14	1	0	1	S		T	1432	28156	78
	15	1	0	1	T		T	1435	27147	73
	16	0	1	0		T		1448	27827	90
	17	1	0	1	S		S	1449	28818	86
	18	1	1	0	S	S		1465	25545	83
	19	1	1	0	S	T		1466	24569	76
	20	1	1	0	T	S		1466	25065	81
	21	0	1	0		S		1468	29790	104
	22	1	1	0	T	T		1469	24209	75
	23	1	0	0	T			1563	32507	132
	24	1	0	0	S			1571	34080	142
	25	0	0	1			T	2138	37386	116
	26	0	0	1			S	2177	38223	126
	27	0	0	0				2704	48584	232

Other Possible Questions

- What combinations (and intensity levels) of development, verification, and validation techniques should be applied in a given context to achieve defined time, quality or cost goals?
- What staffing levels should be assigned to achieve time, quality or cost targets?
- Does investment in training pay off for specific development contexts and goals?
- Do investments in improving development, verification, and validation techniques pay off for specific development contexts and goals?
- What are promising areas of research for improving development, verification, and validation techniques?
- …

Thank You