Presentation at Software Technology Forum
Von Neumann Computer Society

development
think
process
©YA2004

Continuous Stretching of your Software Engineering Capability

A System Thinking for SW Process

Yoshihiro Akiyama (y.akiyama@ieee.org)

Graduate School, Systems & Computer Science, and

Center for Information, Communication, and Technology Education

May 23th,2008

KIT

Kyushu Institute of Technology

Agenda

Software is technology and enabler

Current Software Practice

“Proper Process” for Software

Stretching Individual Capability by PSP

Stretching Team and Management Capability by TSP

Communicating with others

Establishing Global Project Management

Conclusion

Trademarks and Service Marks

- The following are service marks of Carnegie Mellon University.
  - CMMI^{SM}
  - Team Software Process^{SM}
  - TSP^{SM}
  - Personal Software Process^{SM}
  - PSP^{SM}

- The following are registered trademarks of Carnegie Mellon University.
  - Capability Maturity Model®
  - CMM®
  - Capability Maturity Model® Integration
  - CMMI®

Software is Technology and the enabler

Today ….

- *ICT* is rapidly advancing.
- *SOA* for more applications is progressing.
- *Global* project management is demanded but thin and short.
- *Software evolution* & maintenance (legacy) are long tail.
- *Software engineers* need to work on many methods, many applications, and many platforms.
- *High quality* is demanded wherever software is used.

Current Software Industry Performance

- Compared to other industries,
  software performance is sometimes disappointed:
  - Overall architecture is not established in early phase and not clean.
  - Many times of delay must be negotiated.
  - Ship date is rarely met.
  - There are no warranties.
  - Customers must pay significantly for the bugs after shipment.
- Large-scale projects are mostly troubled.

Especially Software Industry wants to say the Quality Objective shoud be

1Defect/KLOC ➔ 1 Defect/MLOC!

However current software quality performance is

- *More than 50% of total efforts is sometimes spent for testing.*
- *Neither safe nor secure software* is produced.
- *Unknown Quality of shipped software* is usual.

“Desired” Proper Process for Software

A step by step to follow in order to produce high quality software <u>consistently</u> as planned:

- Framework for self managing individual,
- Framework for self directed team.
- How to Improve estimating and managing project work is included.

Optimizing

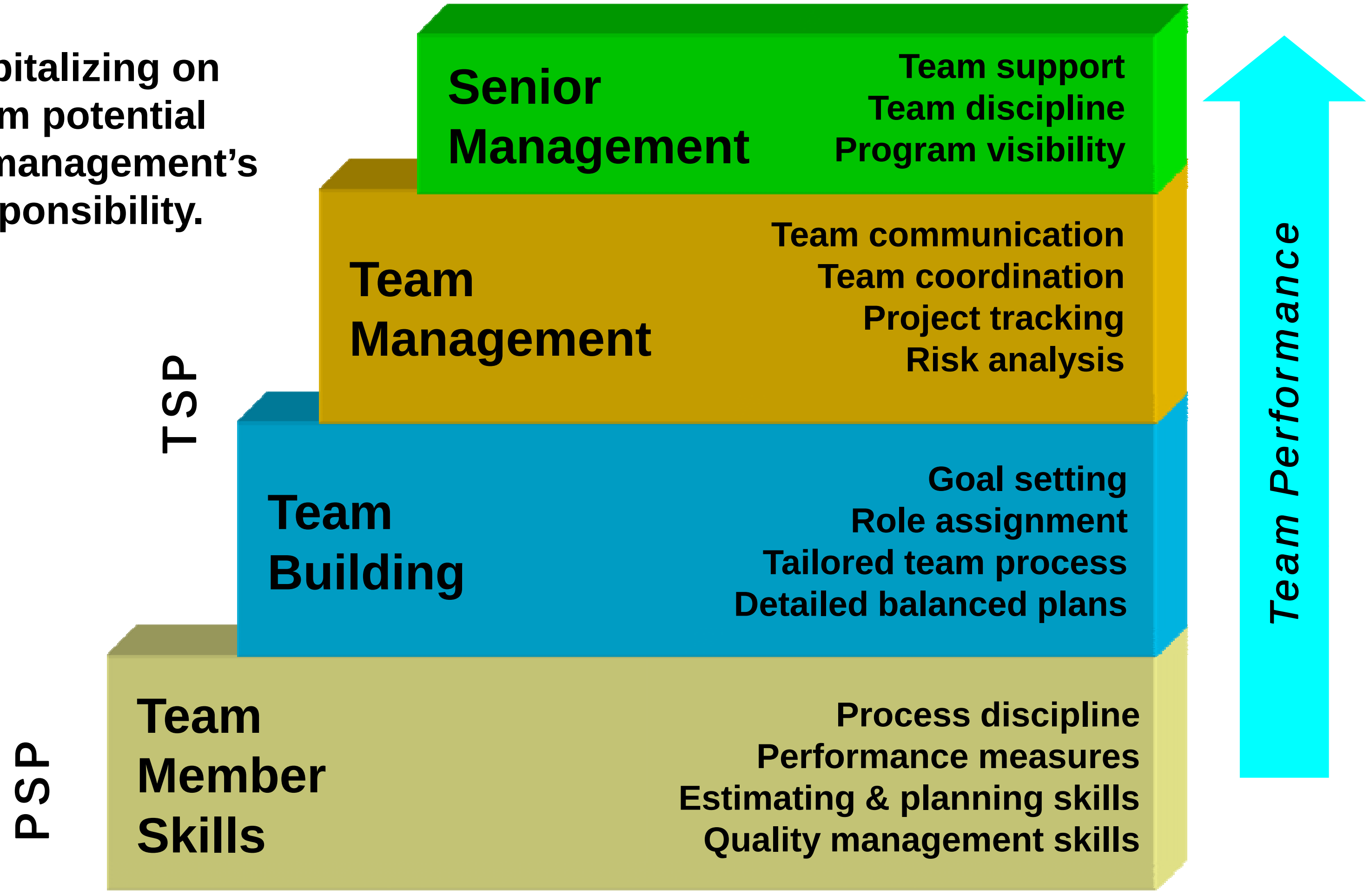
- Project resources, customer satisfaction, and quality
- Everywhere

Scalable

- Works, teams, and locations, users, and resources

Purpose of PSP and TSP: Building High-Performance Teams

Capitalizing on team potential is management's responsibility.



Senior Management

Team support
Team discipline
Program visibility

Team Management

Team communication
Team coordination
Project tracking
Risk analysis

Team Building

Goal setting
Role assignment
Tailored team process
Detailed balanced plans

Team Member Skills

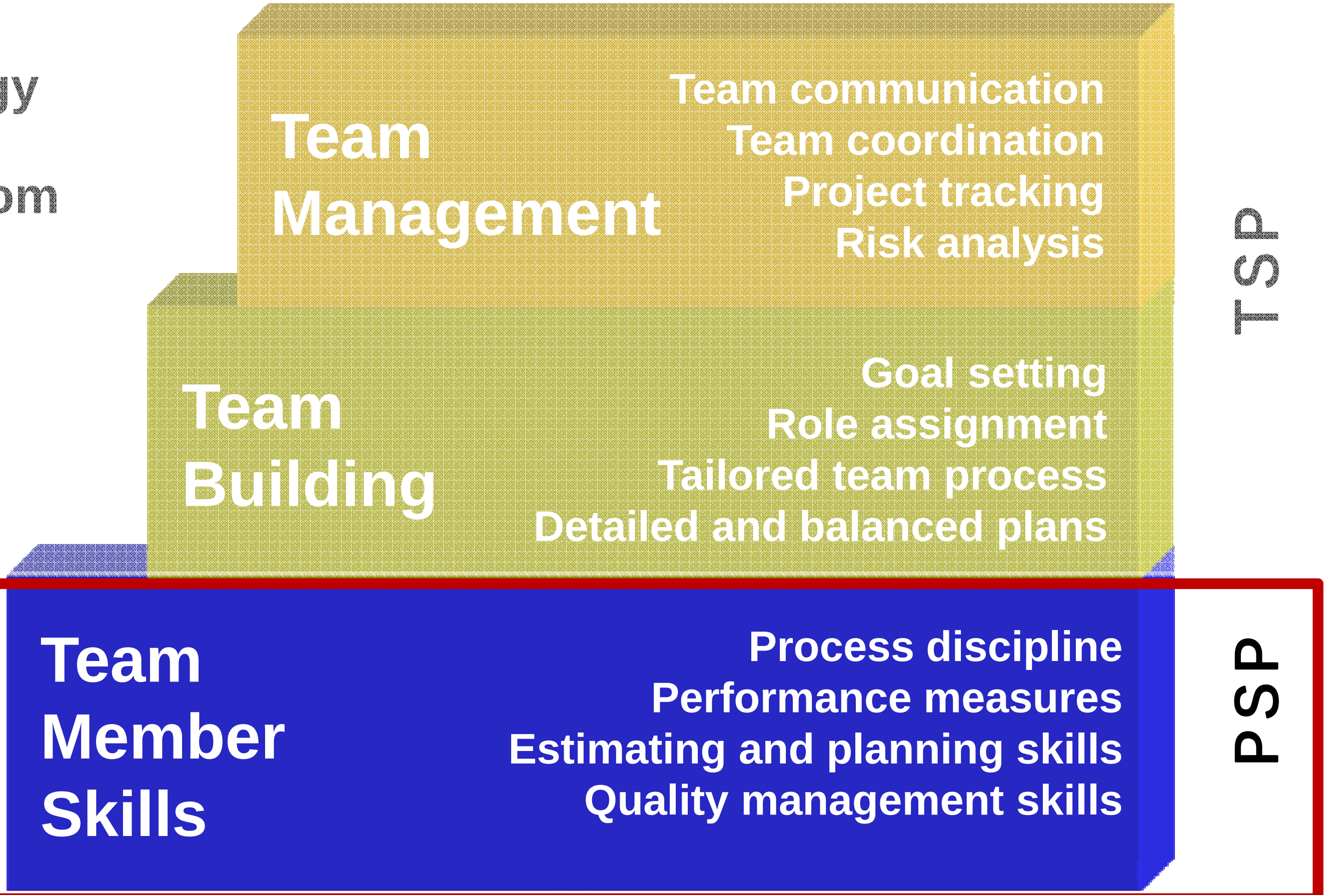
Process discipline
Performance measures
Estimating & planning skills
Quality management skills

Reference: CMU/SEI's course "Managing TSP Teams"

Build High-Performance Individuals

- The TSP strategy is to improve performance from the bottom up.

- **This strategy starts with PSP training.**



Team Management

Team communication
Team coordination
Project tracking
Risk analysis

Team Building

Goal setting
Role assignment
Tailored team process
Detailed and balanced plans

Team Member Skills

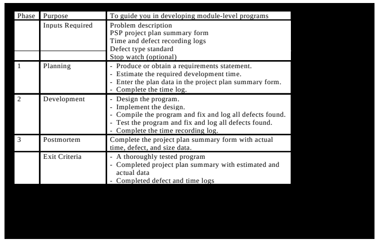
Process discipline
Performance measures
Estimating and planning skills
Quality management skills

TSP

PSP

Reference: CMU/SEI's course “Managing TSP Teams”

The Process Elements



Phase	Purpose	To guide you in developing module-level programs
	Inputs Required	Problem description PSP project plan summary form Time and defect recording logs Defect type standard Stop watch (optional)
1	Planning	- Produce or obtain a requirements statement. - Estimate the required development time. - Enter the plan data in the project plan summary form. - Complete the time log.
2	Development	- Design the program. - Implement the design. - Compile the program and fix and log all defects found. - Test the program and fix and log all defects found. - Complete the time recording log.
3	Postmortem	Complete the project plan summary form with actual time, defect, and size data.
	Exit Criteria	- A thoroughly tested program - Completed project plan summary with estimated and actual data - Completed defect and time logs

Scripts — **Document the process entry criteria, phases/ steps, and exit criteria. The purpose is to guide users of the process.**

Measures — **Measure the process and the product. They provide insight into how the process is working and the status of the work.**

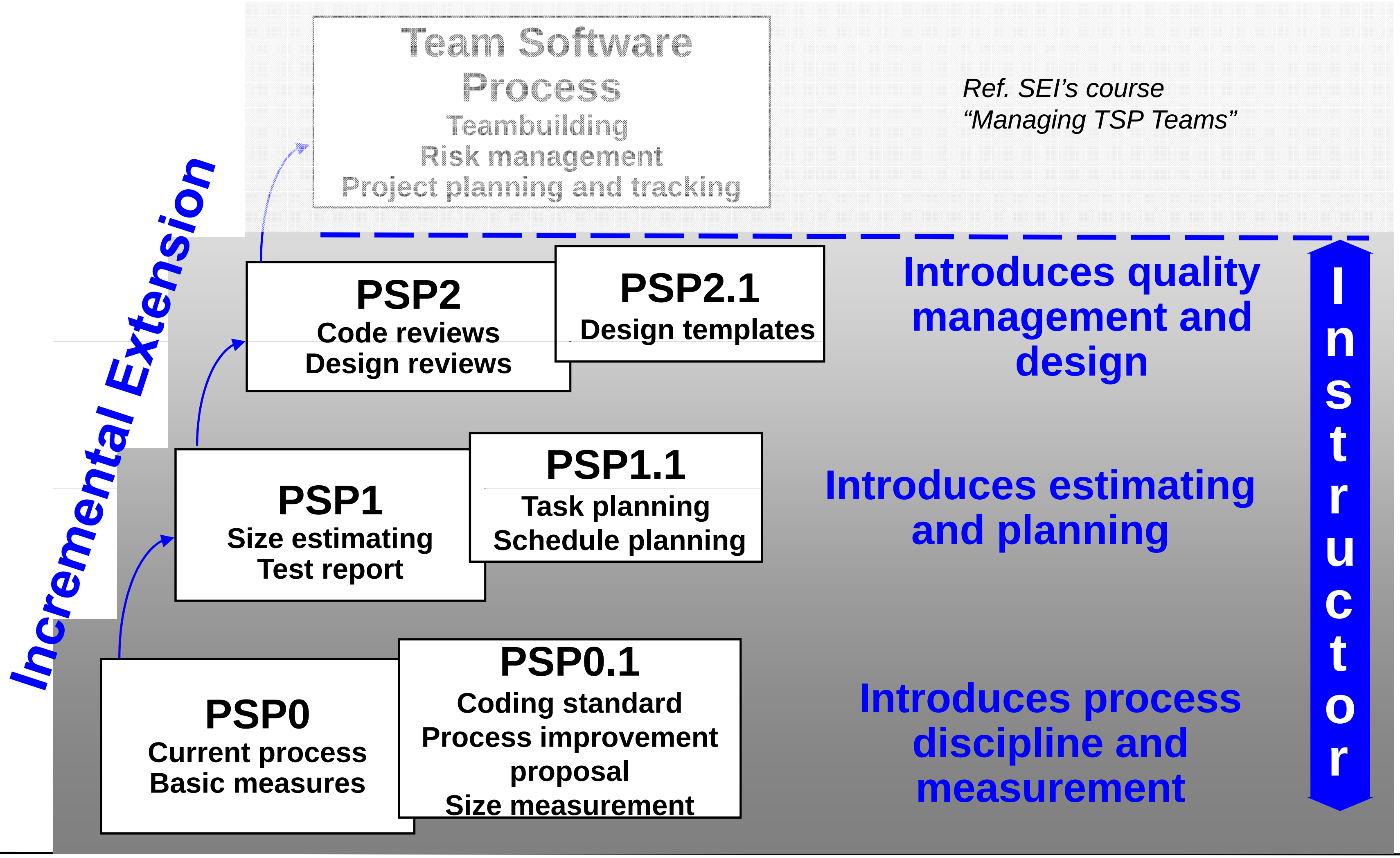
Forms — **Provide a convenient and consistent framework for gathering and retaining data**

Standards — **Provide consistent definitions that guide the work and gathering of data.**

Tools — **Provide automated accepting, handling, processing, and visualizing process data**

Ref. Don Burton, “Introduction to PSP and TSP, SEPG Conference March 2006

The PSP Process Training Structure



Team Software Process
Teambuilding
Risk management
Project planning and tracking

Ref. SEI's course "Managing TSP Teams"

PSP2
Code reviews
Design reviews

PSP2.1
Design templates

Introduces quality management and design

PSP1
Size estimating
Test report

PSP1.1
Task planning
Schedule planning

Introduces estimating and planning

PSP0
Current process
Basic measures

PSP0.1
Coding standard
Process improvement proposal
Size measurement

Introduces process discipline and measurement

Incremental Extension

Instructor

PSP Process Script

©2005 SEI TSP

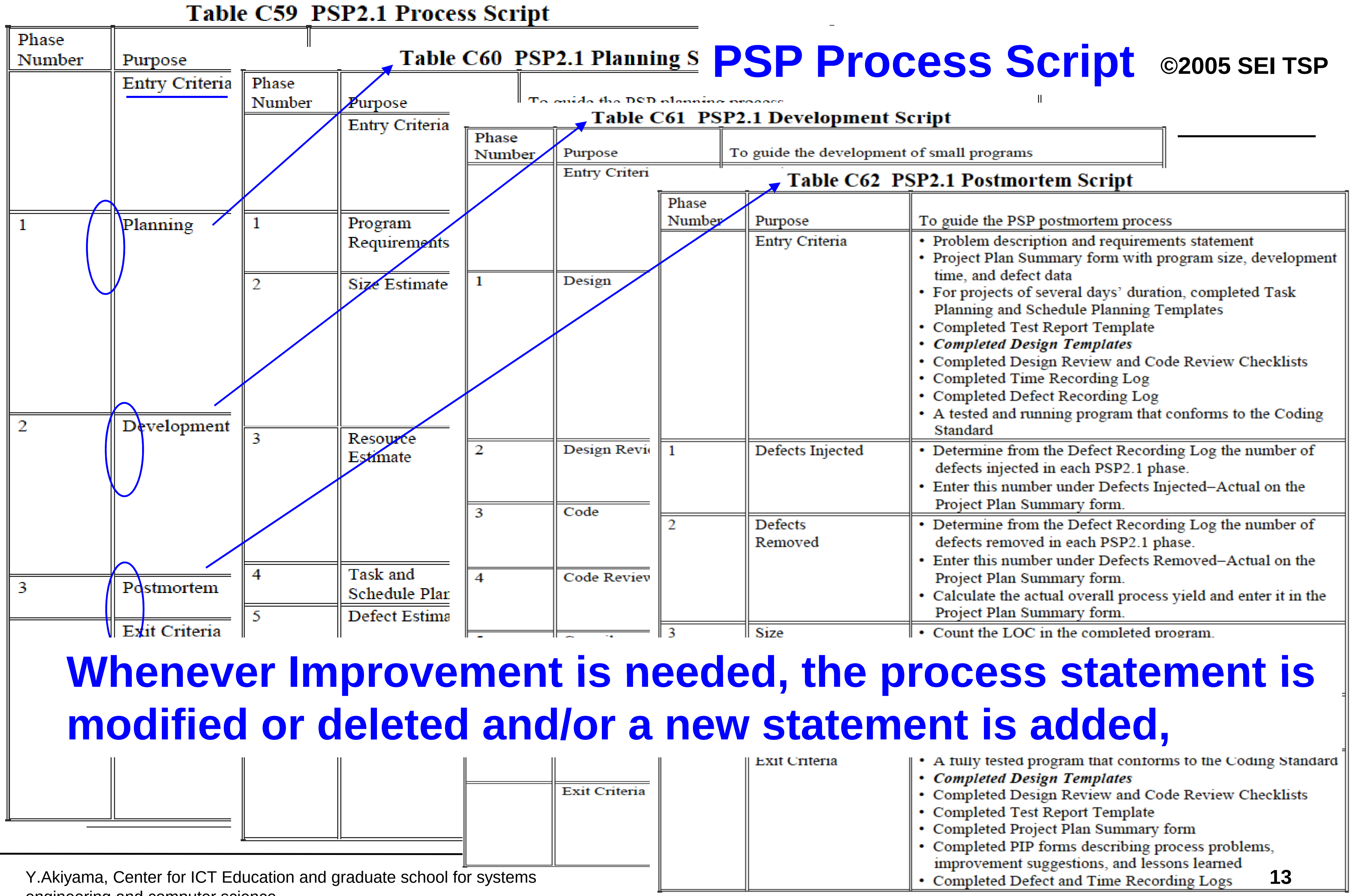


Table C59 PSP2.1 Process Script

Phase Number	Purpose
	Entry Criteria
1	Planning
2	Development
3	Postmortem
	Exit Criteria

Table C60 PSP2.1 Planning S

Phase Number	Purpose	To guide the PSP planning process
	Entry Criteria	
1	Program Requirements	
2	Size Estimate	
3	Resource Estimate	
4	Task and Schedule Plar	
5	Defect Estima	

Table C61 PSP2.1 Development Script

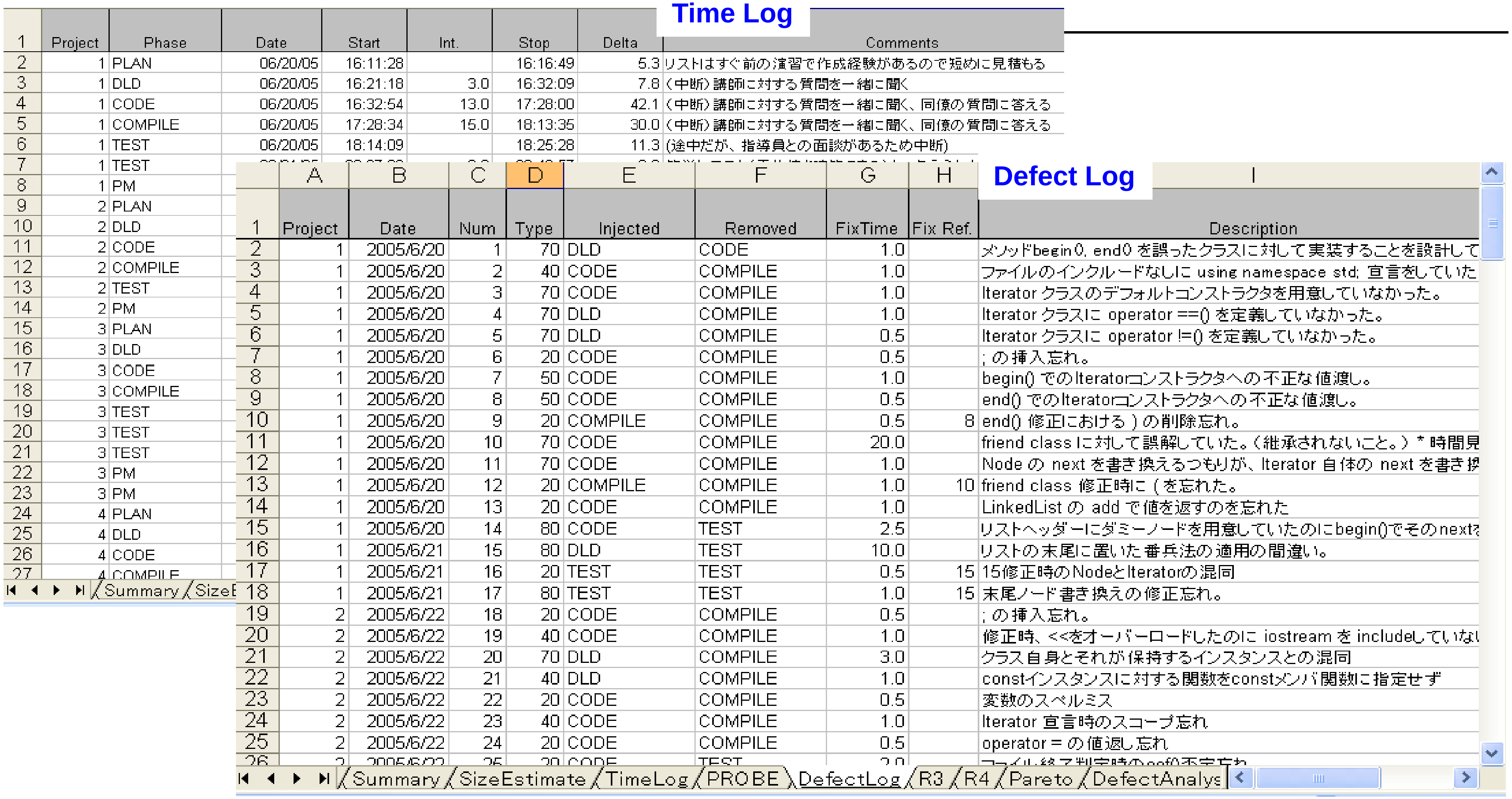
Phase Number	Purpose	To guide the development of small programs
	Entry Criteri	
1	Design	
2	Design Revi	
3	Code	
4	Code Reviev	
	Exit Criteria	

Table C62 PSP2.1 Postmortem Script

Phase Number	Purpose	To guide the PSP postmortem process
	Entry Criteria	• Problem description and requirements statement • Project Plan Summary form with program size, development time, and defect data • For projects of several days' duration, completed Task Planning and Schedule Planning Templates • Completed Test Report Template • ***Completed Design Templates*** • Completed Design Review and Code Review Checklists • Completed Time Recording Log • Completed Defect Recording Log • A tested and running program that conforms to the Coding Standard
1	Defects Injected	• Determine from the Defect Recording Log the number of defects injected in each PSP2.1 phase. • Enter this number under Defects Injected–Actual on the Project Plan Summary form.
2	Defects Removed	• Determine from the Defect Recording Log the number of defects removed in each PSP2.1 phase. • Enter this number under Defects Removed–Actual on the Project Plan Summary form. • Calculate the actual overall process yield and enter it in the Project Plan Summary form.
3	Size	• Count the LOC in the completed program.
	Exit Criteria	• A fully tested program that conforms to the Coding Standard • ***Completed Design Templates*** • Completed Design Review and Code Review Checklists • Completed Test Report Template • Completed Project Plan Summary form • Completed PIP forms describing process problems, improvement suggestions, and lessons learned • Completed Defect and Time Recording Logs

Whenever Improvement is needed, the process statement is modified or deleted and/or a new statement is added,

PSP Basic Input



Time Log

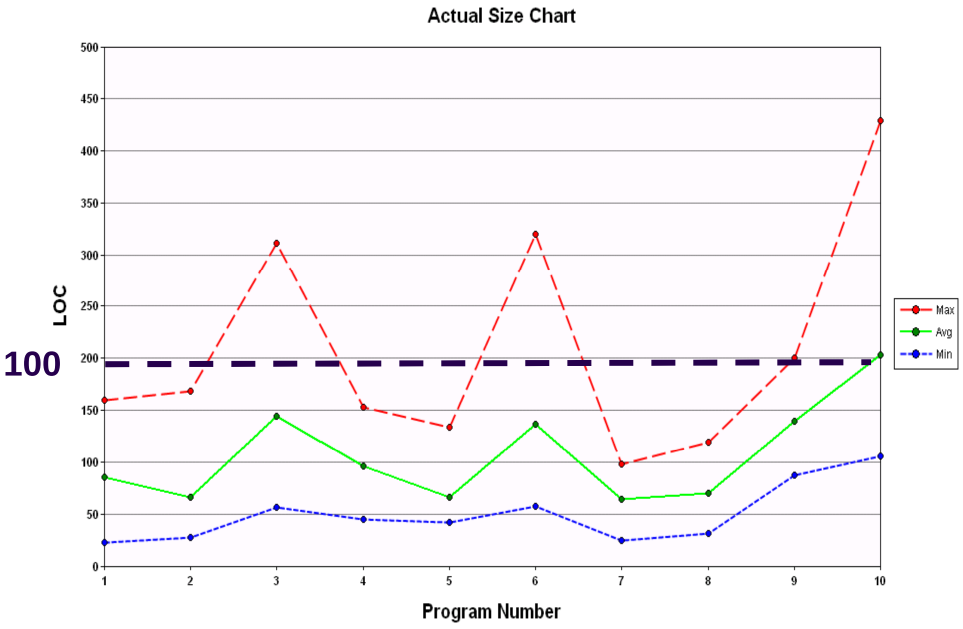
1	Project	Phase	Date	Start	Int.	Stop	Delta	Comments
2	1	PLAN	06/20/05	16:11:28		16:16:49	5.3	リストはすぐ前の演習で作成経験があるので短めに見積もる
3	1	DLD	06/20/05	16:21:18	3.0	16:32:09	7.8	(中断) 講師に対する質問を一緒に聞く
4	1	CODE	06/20/05	16:32:54	13.0	17:28:00	42.1	(中断) 講師に対する質問を一緒に聞く、同僚の質問に答える
5	1	COMPILE	06/20/05	17:28:34	15.0	18:13:35	30.0	(中断) 講師に対する質問を一緒に聞く、同僚の質問に答える
6	1	TEST	06/20/05	18:14:09		18:25:28	11.3	(途中だが、指導員との面談があるため中断)
7	1	TEST						
8	1	PM						
9	2	PLAN						
10	2	DLD						
11	2	CODE						
12	2	COMPILE						
13	2	TEST						
14	2	PM						
15	3	PLAN						
16	3	DLD						
17	3	CODE						
18	3	COMPILE						
19	3	TEST						
20	3	TEST						
21	3	TEST						
22	3	PM						
23	3	PM						
24	4	PLAN						
25	4	DLD						
26	4	CODE						
27	4	COMPILE						

Defect Log

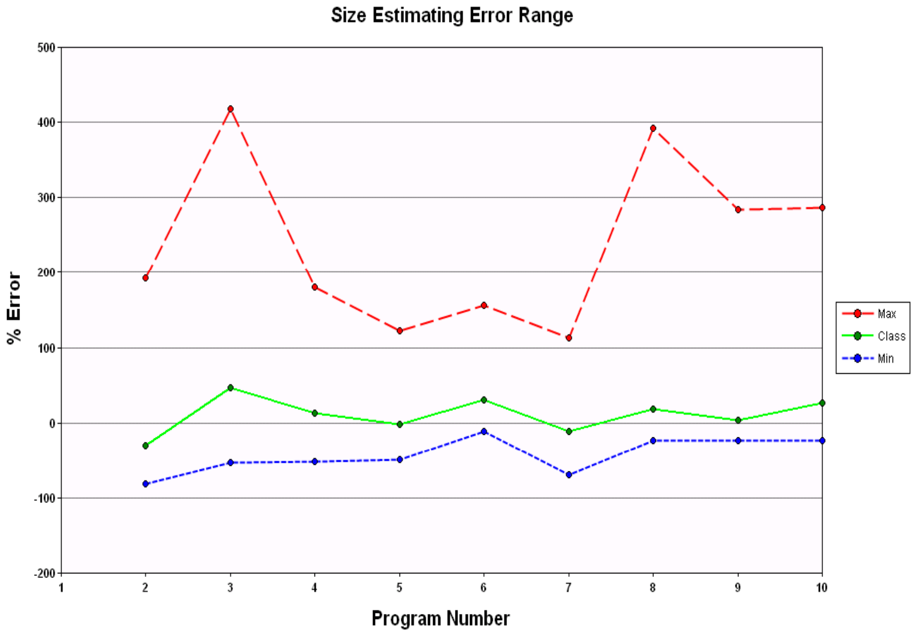
	A	B	C	D	E	F	G	H	I
1	Project	Date	Num	Type	Injected	Removed	FixTime	Fix Ref.	Description
2	1	2005/6/20	1	70	DLD	CODE	1.0		メソッドbegin(), end() を誤ったクラスに対して実装することを設計して
3	1	2005/6/20	2	40	CODE	COMPILE	1.0		ファイルのインクルードなしに using namespace std; 宣言をしていた
4	1	2005/6/20	3	70	CODE	COMPILE	1.0		Iterator クラスのデフォルトコンストラクタを用意していなかった。
5	1	2005/6/20	4	70	DLD	COMPILE	1.0		Iterator クラスに operator ==() を定義していなかった。
6	1	2005/6/20	5	70	DLD	COMPILE	0.5		Iterator クラスに operator !=() を定義していなかった。
7	1	2005/6/20	6	20	CODE	COMPILE	0.5		; の挿入忘れ。
8	1	2005/6/20	7	50	CODE	COMPILE	1.0		begin() でのIteratorコンストラクタへの不正な値渡し。
9	1	2005/6/20	8	50	CODE	COMPILE	0.5		end() でのIteratorコンストラクタへの不正な値渡し。
10	1	2005/6/20	9	20	COMPILE	COMPILE	0.5	8	end() 修正における) の削除忘れ。
11	1	2005/6/20	10	70	CODE	COMPILE	20.0		friend class に対して誤解していた。(継承されないこと。) * 時間見
12	1	2005/6/20	11	70	CODE	COMPILE	1.0		Node の next を書き換えるつもりが、Iterator 自体の next を書き換
13	1	2005/6/20	12	20	COMPILE	COMPILE	1.0	10	friend class 修正時に (を忘れた。
14	1	2005/6/20	13	20	CODE	COMPILE	1.0		LinkedList の add で値を返すのを忘れた
15	1	2005/6/20	14	80	CODE	TEST	2.5		リストヘッダーにダミーノードを用意していたのにbegin()でそのnextを
16	1	2005/6/21	15	80	DLD	TEST	10.0		リストの末尾に置いた番兵法の適用の間違い。
17	1	2005/6/21	16	20	TEST	TEST	0.5	15	15修正時のNodeとIteratorの混同
18	1	2005/6/21	17	80	TEST	TEST	1.0	15	末尾ノード書き換えの修正忘れ。
19	2	2005/6/22	18	20	CODE	COMPILE	0.5		; の挿入忘れ。
20	2	2005/6/22	19	40	CODE	COMPILE	1.0		修正時、<<をオーバーロードしたのに iostream を includeしていない
21	2	2005/6/22	20	70	DLD	COMPILE	3.0		クラス自身とそれが保持するインスタンスとの混同
22	2	2005/6/22	21	40	DLD	COMPILE	1.0		constインスタンスに対する関数をconstメンバ関数に指定せず
23	2	2005/6/22	22	20	CODE	COMPILE	0.5		変数のスペルミス
24	2	2005/6/22	23	40	CODE	COMPILE	1.0		Iterator 宣言時のスコープ忘れ
25	2	2005/6/22	24	20	CODE	COMPILE	0.5		operator = の値返し忘れ

PSP Estimate Accuracy

Size

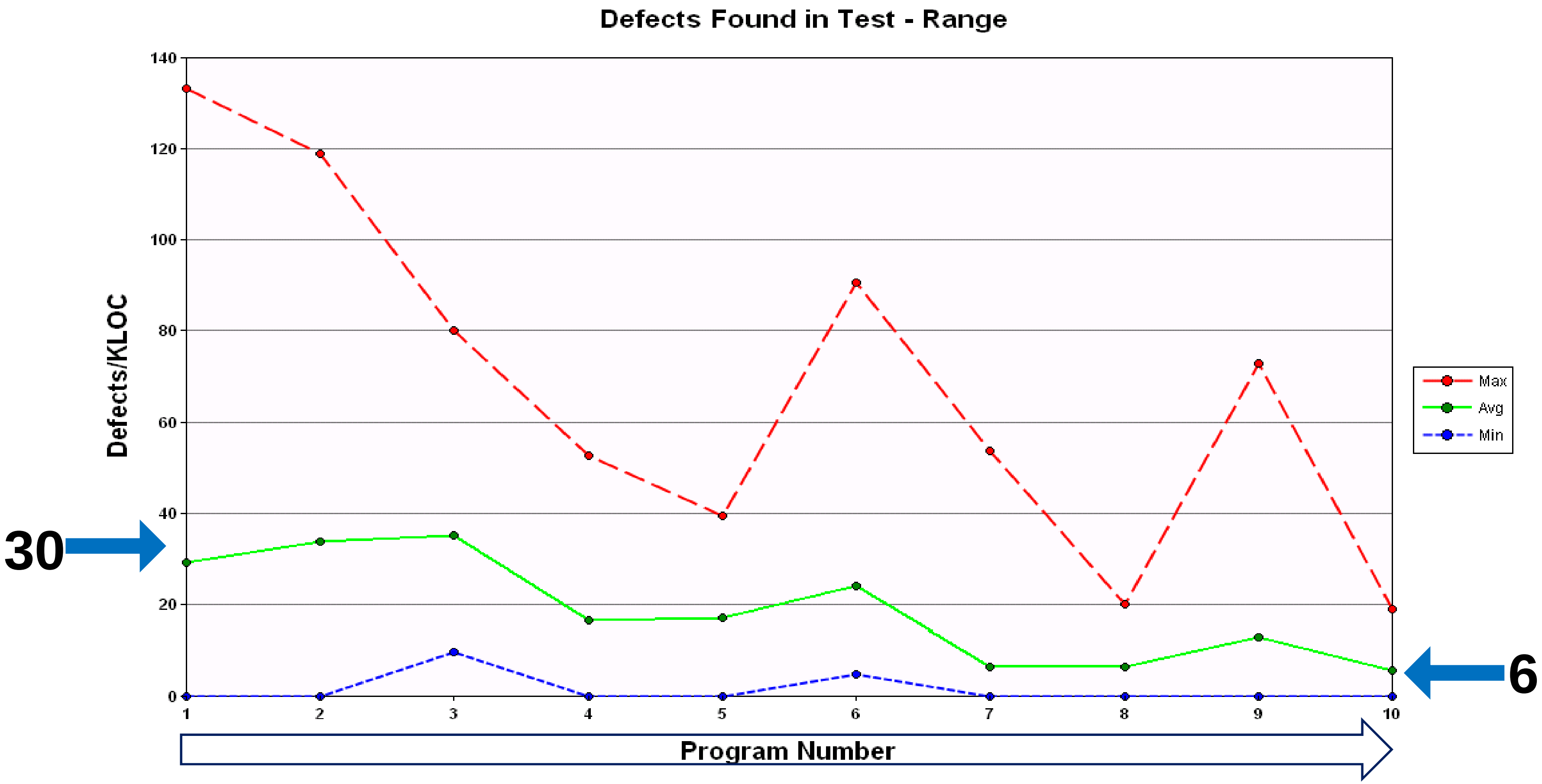


% Error in Size Estimate



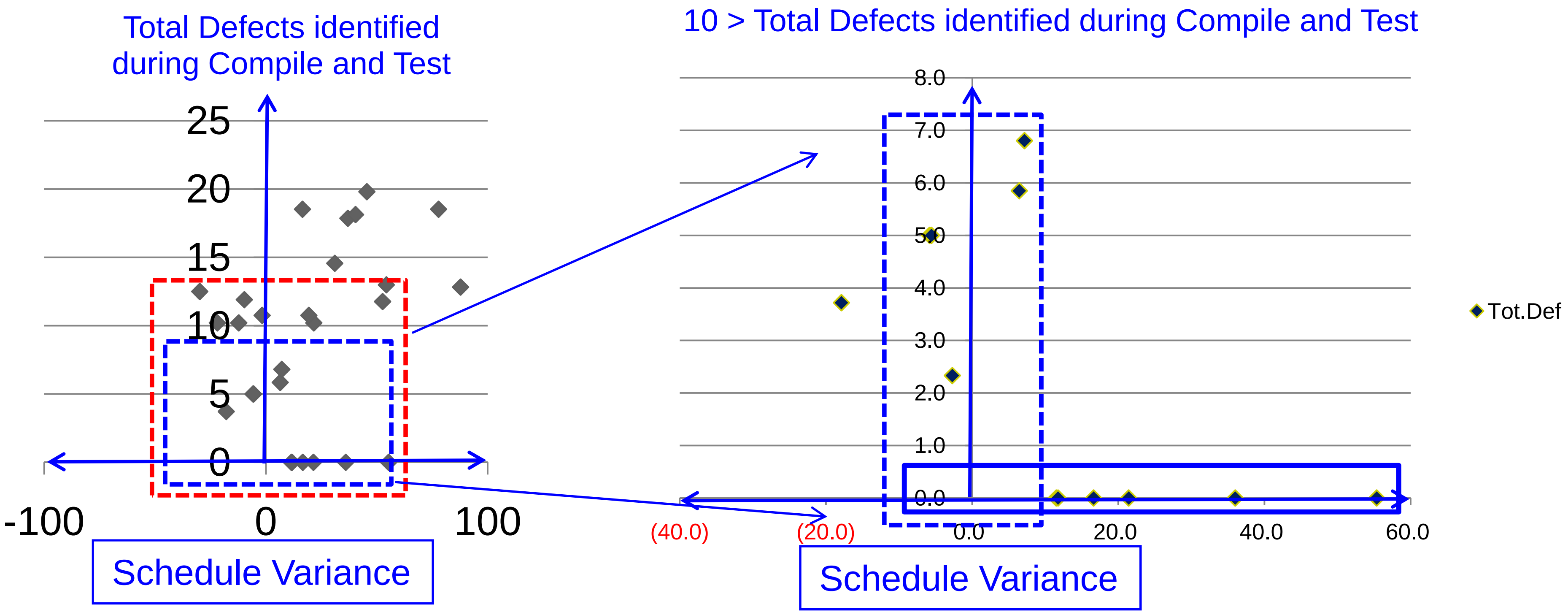
PSP – Quality Improvement

Test defects found **1/5** of the original defect amount



PSP Predictability on Schedule

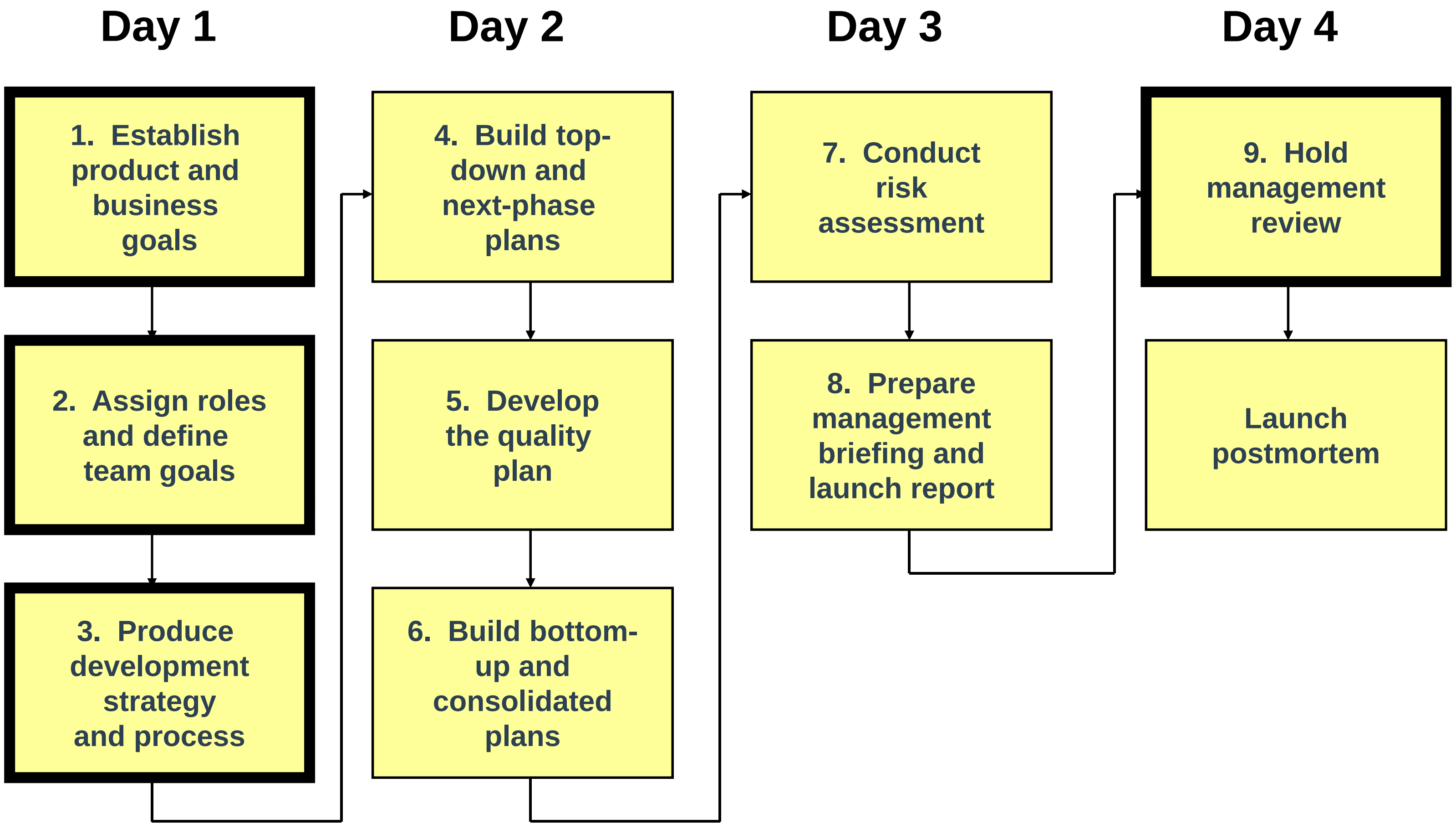
Quality is the major control parameter for the variance



About 10% variation identified for 0<# of defects identified for compile and test <10.
Large (60%) variation allowed for 0 = zero compile and test defects

Ref.: Yoshihiro Akiyama, Who could be Teacher for High Quality Software in Special Session: Preparing Students for Industry’s Software Engineering needs, organized by Watts Humphrey, CSEE&T 2008

The Launch Process Meetings



Day 1

1. Establish product and business goals

2. Assign roles and define team goals

3. Produce development strategy and process

Day 2

4. Build top-down and next-phase plans

5. Develop the quality plan

6. Build bottom-up and consolidated plans

Day 3

7. Conduct risk assessment

8. Prepare management briefing and launch report

Day 4

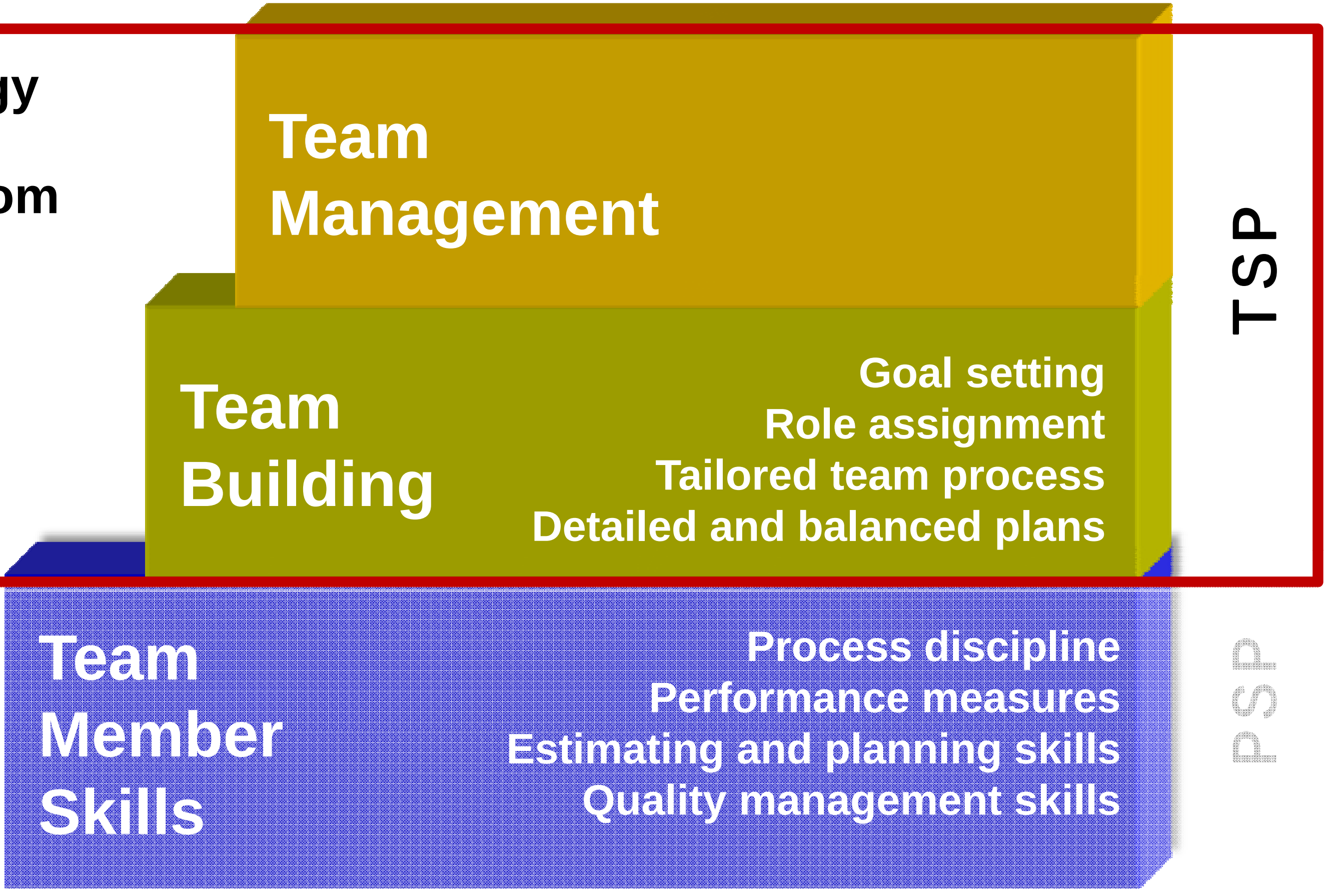
9. Hold management review

Launch postmortem

Reference: CMU/SEI's course "Managing TSP Teams"

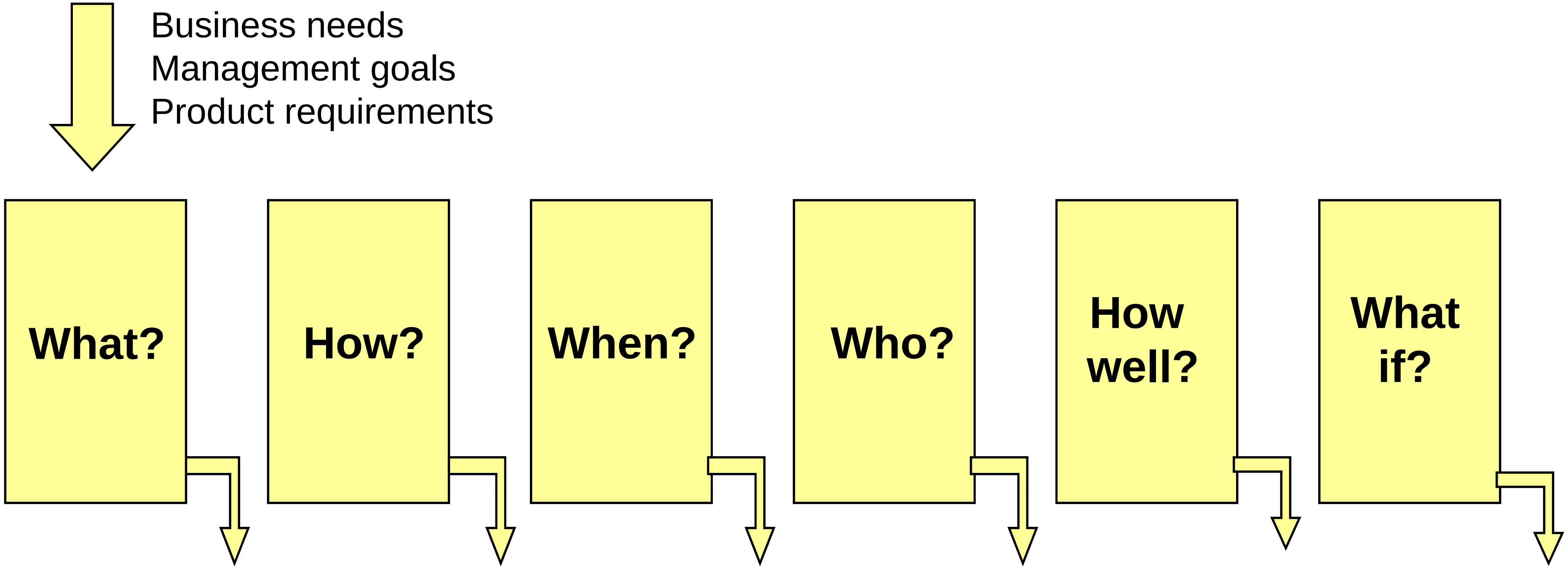
Build and Maintain High-Performance Teams

- The TSP strategy is to improve performance from the bottom up.
- This strategy starts with PSP training.



Reference: CMU/SEI's course "Managing TSP Teams"

The TSP Launch Products



Business needs
Management goals
Product requirements

What?

How?

When?

Who?

How well?

What if?

- Team goals
- **Conceptual design**
- **Planned products**
- Size estimates

- **Team strategy**
- Team defined process

- Task plan
- Schedule Plan
- Earned-value Plan

- Team roles
- Task plans
- Earned-value Plan

- **Quality plan**

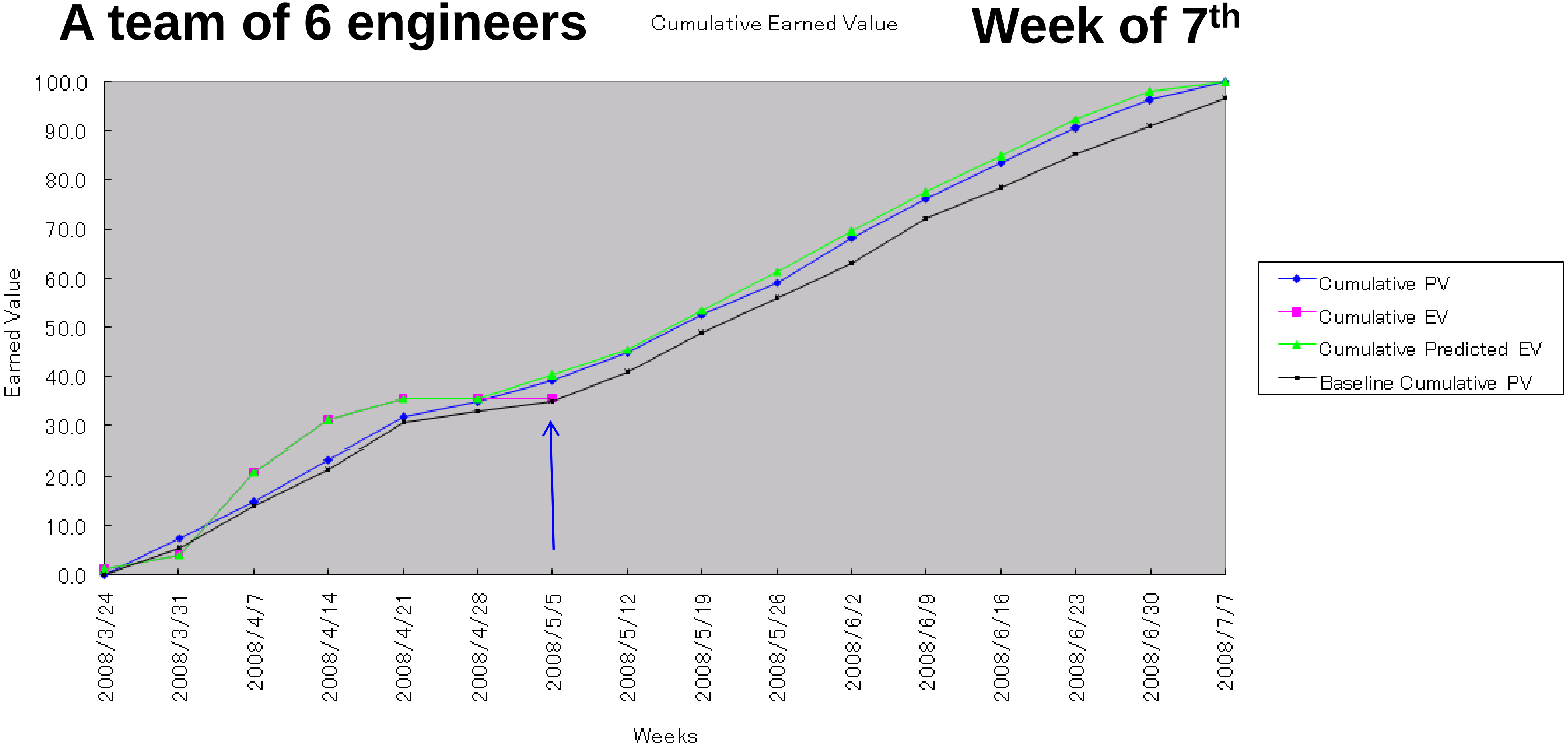
- Risk evaluation
- Alternate plans

Reference: CMU/SEI’s course “Managing TSP Teams”

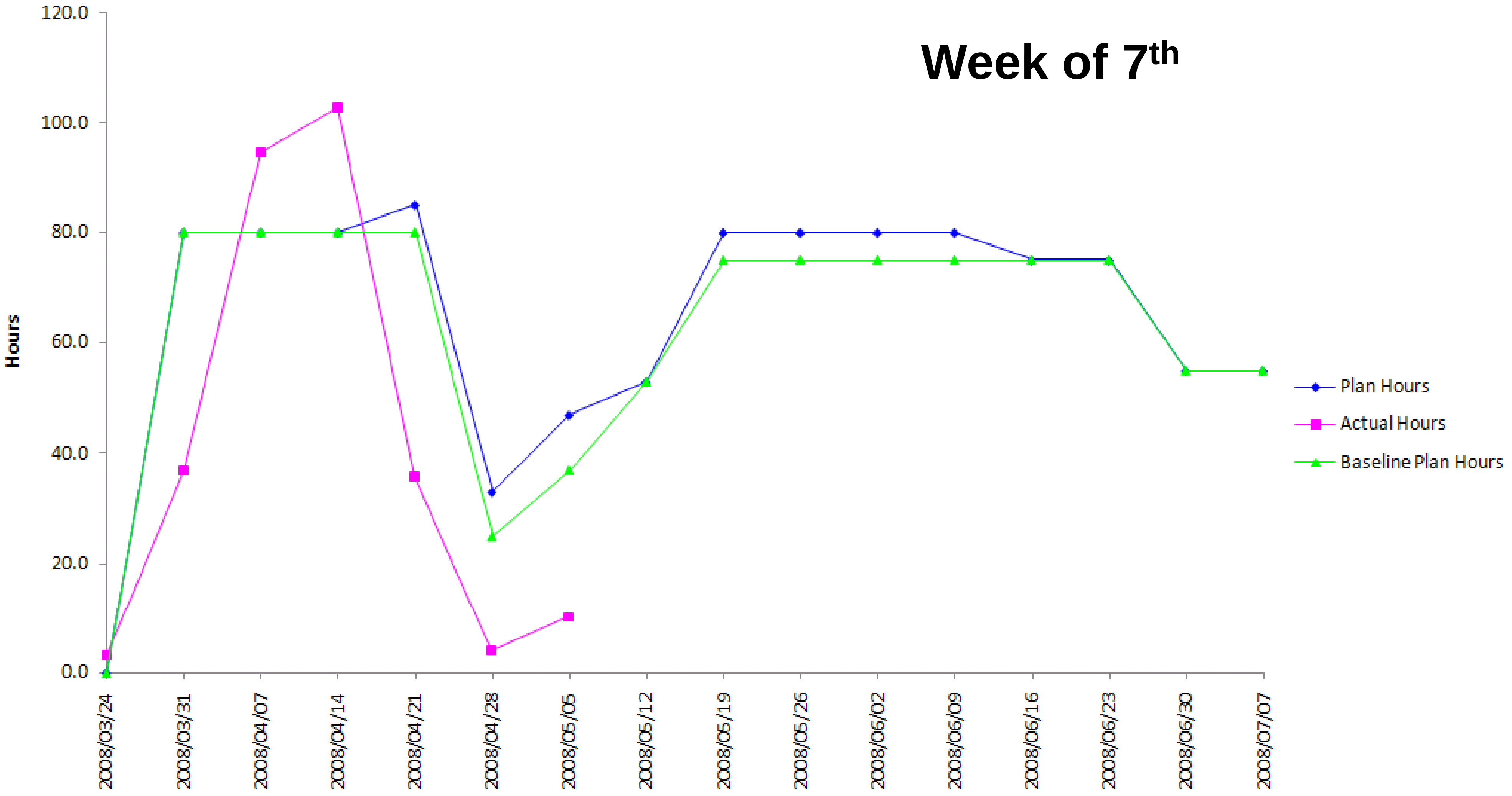
Progress – Accumulated Earned Value

A team of 6 engineers

Week of 7th



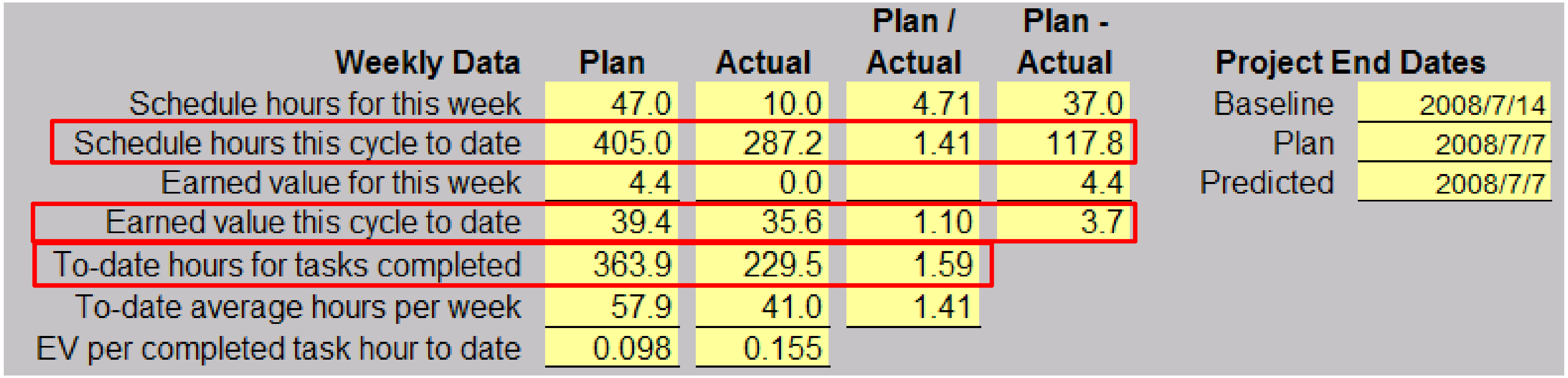
Week Task Hours - Plan vs. Actual



Team Communication

Plan / estimation accuracy

Week of 7th



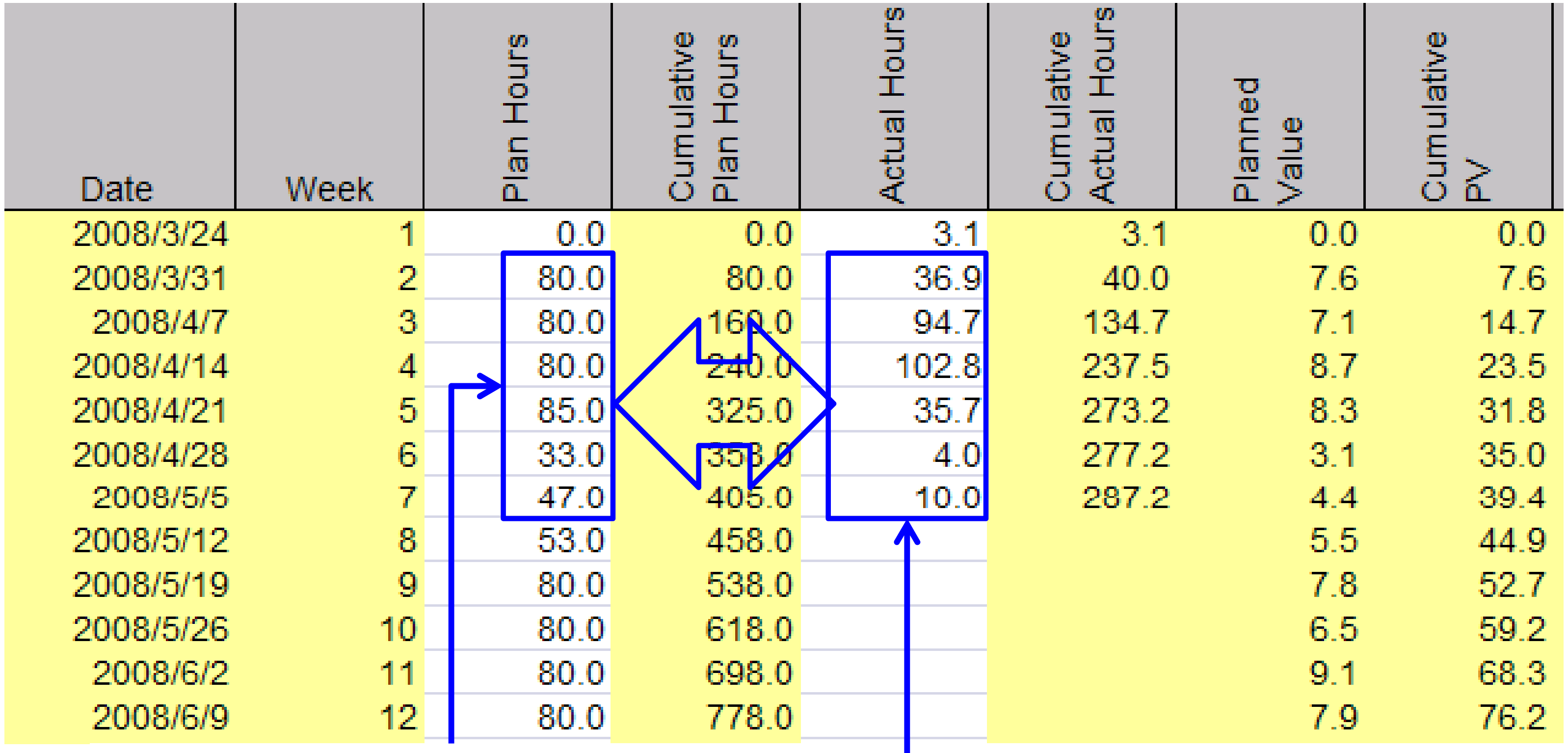
Weekly Data	Plan	Actual	Plan / Actual	Plan - Actual	Project End Dates	
Schedule hours for this week	47.0	10.0	4.71	37.0	Baseline	2008/7/14
Schedule hours this cycle to date	405.0	287.2	1.41	117.8	Plan	2008/7/7
Earned value for this week	4.4	0.0		4.4	Predicted	2008/7/7
Earned value this cycle to date	39.4	35.6	1.10	3.7		
To-date hours for tasks completed	363.9	229.5	1.59			
To-date average hours per week	57.9	41.0	1.41			
EV per completed task hour to date	0.098	0.155				

- To-date EV is 3.8% below plan (39.4 vs. 35.6).
- Effort has been overestimated by 59%. Is this a trend?
- 57.7 hours (287.2 – 229.5) have been spent on incomplete tasks.

The estimation is not accurate (59% over estimate).
The progress is on track.

Acquisition of Task Hours

Week of 7th



Date	Week	Plan Hours	Cumulative Plan Hours	Actual Hours	Cumulative Actual Hours	Planned Value	Cumulative PV
2008/3/24	1	0.0	0.0	3.1	3.1	0.0	0.0
2008/3/31	2	80.0	80.0	36.9	40.0	7.6	7.6
2008/4/7	3	80.0	160.0	94.7	134.7	7.1	14.7
2008/4/14	4	80.0	240.0	102.8	237.5	8.7	23.5
2008/4/21	5	85.0	325.0	35.7	273.2	8.3	31.8
2008/4/28	6	33.0	358.0	4.0	277.2	3.1	35.0
2008/5/5	7	47.0	405.0	10.0	287.2	4.4	39.4
2008/5/12	8	53.0	458.0			5.5	44.9
2008/5/19	9	80.0	538.0			7.8	52.7
2008/5/26	10	80.0	618.0			6.5	59.2
2008/6/2	11	80.0	698.0			9.1	68.3
2008/6/9	12	80.0	778.0			7.9	76.2

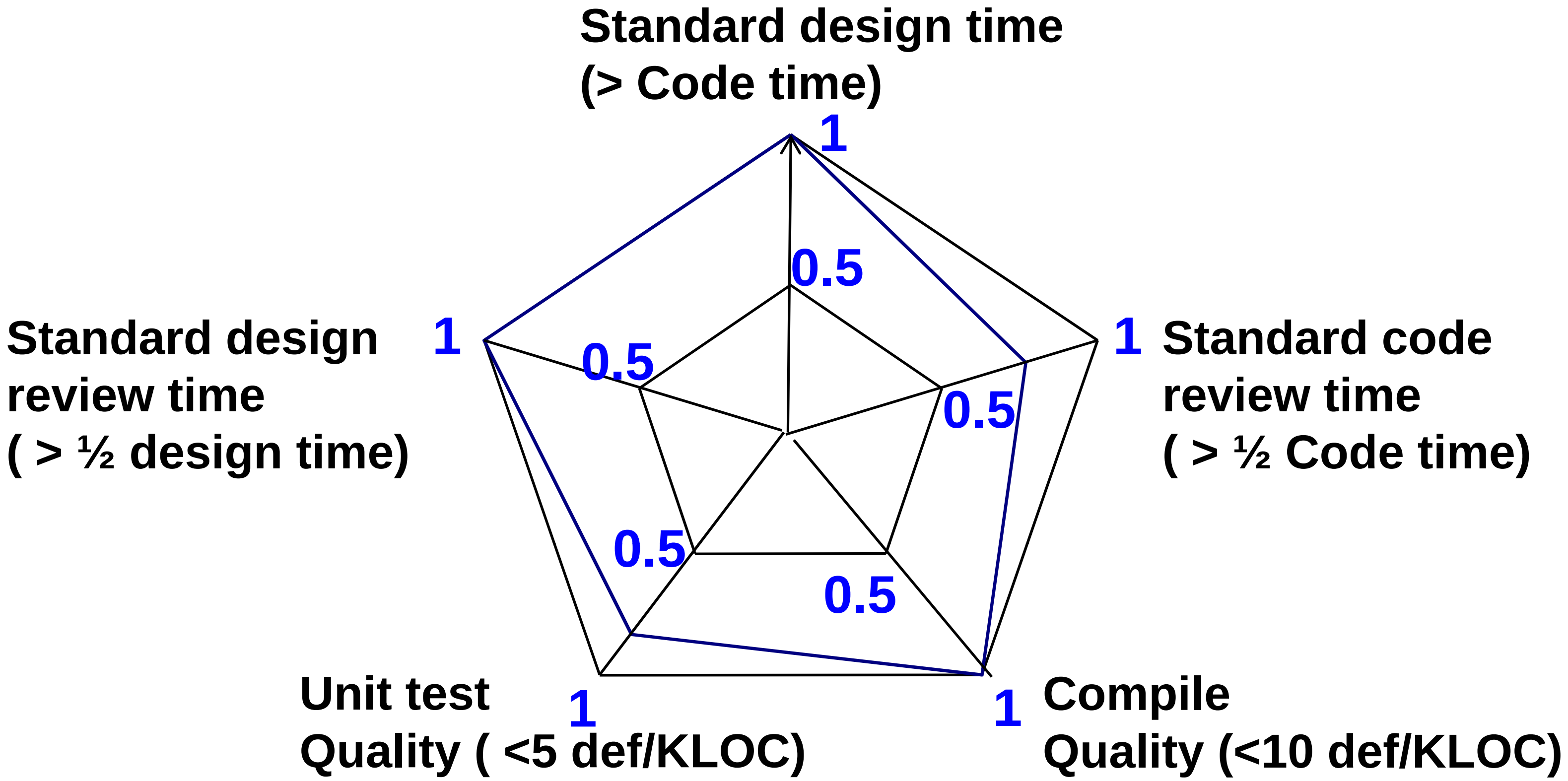
Average 68 hours/week/team
➔ Average 11 hours/week/per.

Average 48 hours/week/team
➔ Average 8 hours/week/per.

Quality Management

- With the TSP, the developers
  - record all of their defects
  - use process data to analyze product quality
  - *strive to fix all defects before test*
- In managing quality, TSP teams use the
  - process quality profile
  - process quality index (PQI)

Component Quality Profile

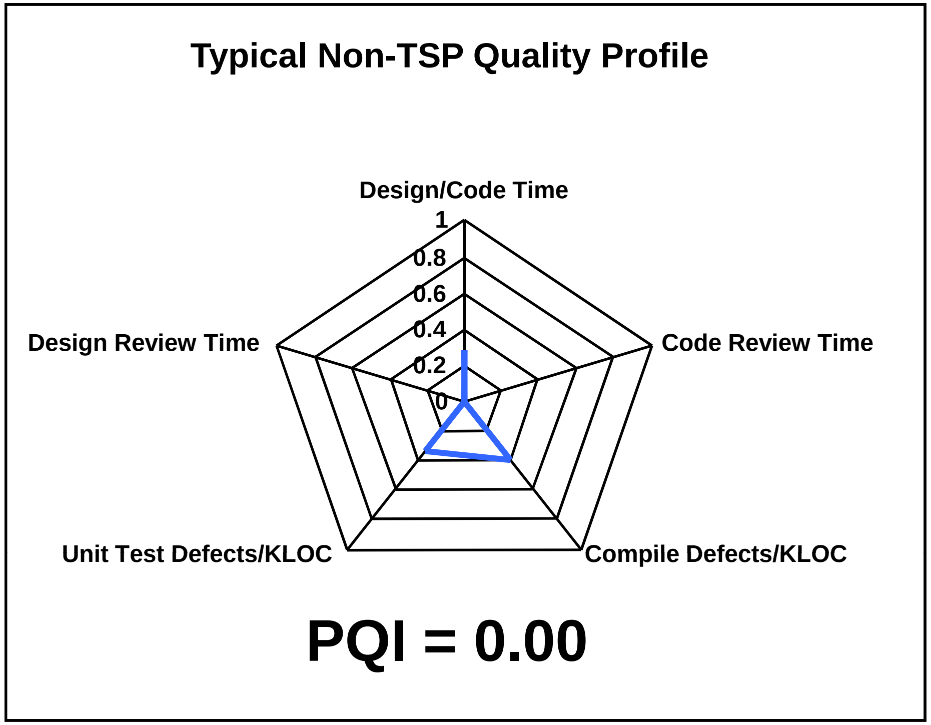


Note: LOC is the measure of Modified and Added Code.

Reference: Watts Humphrey, PSP – A Self-Improvement Process for Software Engineers, Addison Wesley, 2005

A Traditional Quality Profile

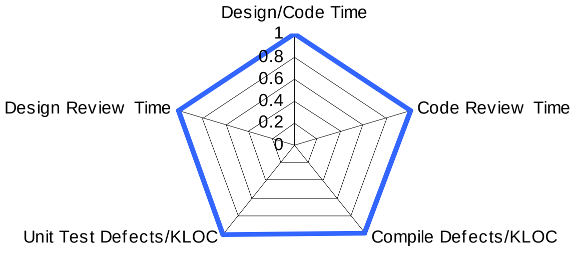
- With current typical software practice, PQI is at or near 0.
- With TSP, PQI is measured and can be managed with control charts.
- No defects have been found when PQI is above 0.4.



Reference: SEI Course “Managing TSP Teams”

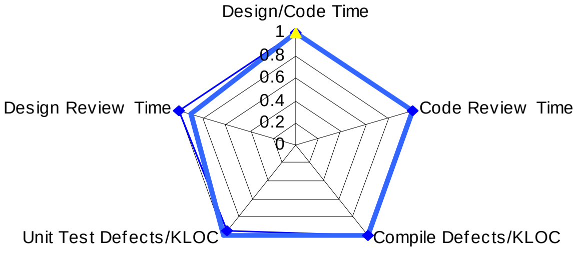
Selected TSP Quality Profiles – before test

Quality Profile for Assembly 1



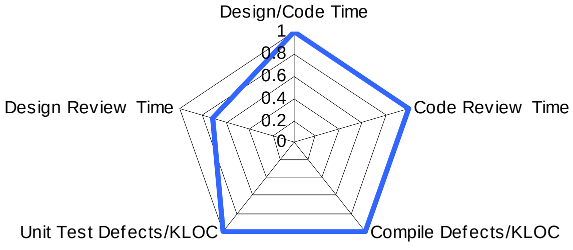
PQI = 0.97

Quality Profile for Assembly 2



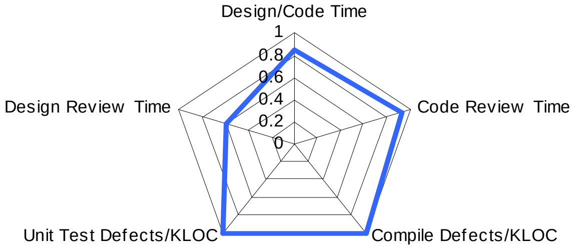
PQI = 0.88

Quality Profile for Assembly 3



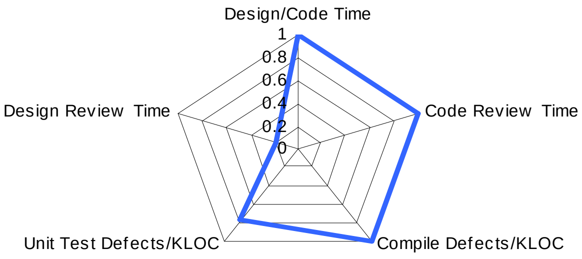
PQI = 0.71

Quality Profile for Assembly 4



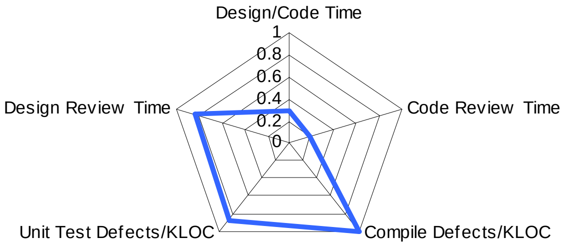
PQI = 0.59

Quality Profile for Assembly 5



PQI = 0.15

Quality Profile for Assembly 6



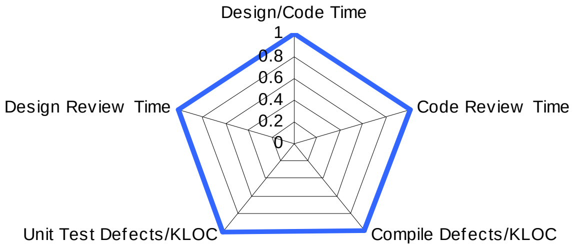
PQI = 0.04

Reference: Watts Humphrey, PSP – A Self-Improvement Process for Software Engineers, Addison Wesley, 2005

Selected TSP Quality Profiles – after test

Quality Profile for Assembly 1

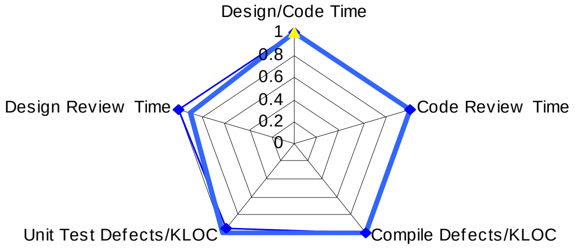
Test defects = 0



PQI = 0.97

Quality Profile for Assembly 2

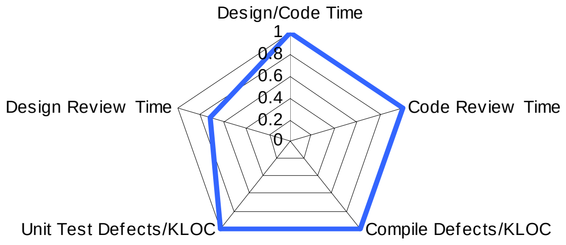
Test defects = 0



PQI = 0.88

Quality Profile for Assembly 3

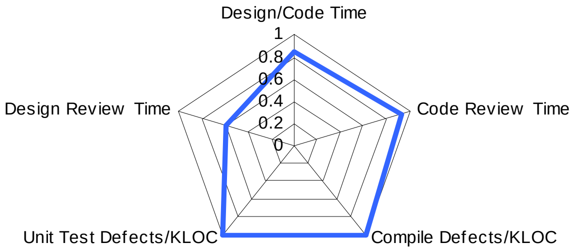
Test defects = 0



PQI = 0.71

Quality Profile for Assembly 4

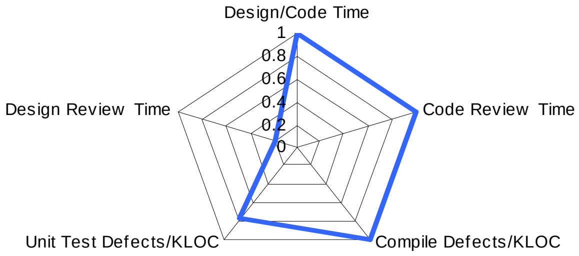
Test defects = 0



PQI = 0.59

Quality Profile for Assembly 5

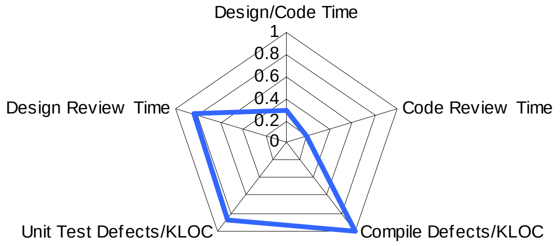
Test defects = 1



PQI = 0.15

Quality Profile for Assembly 6

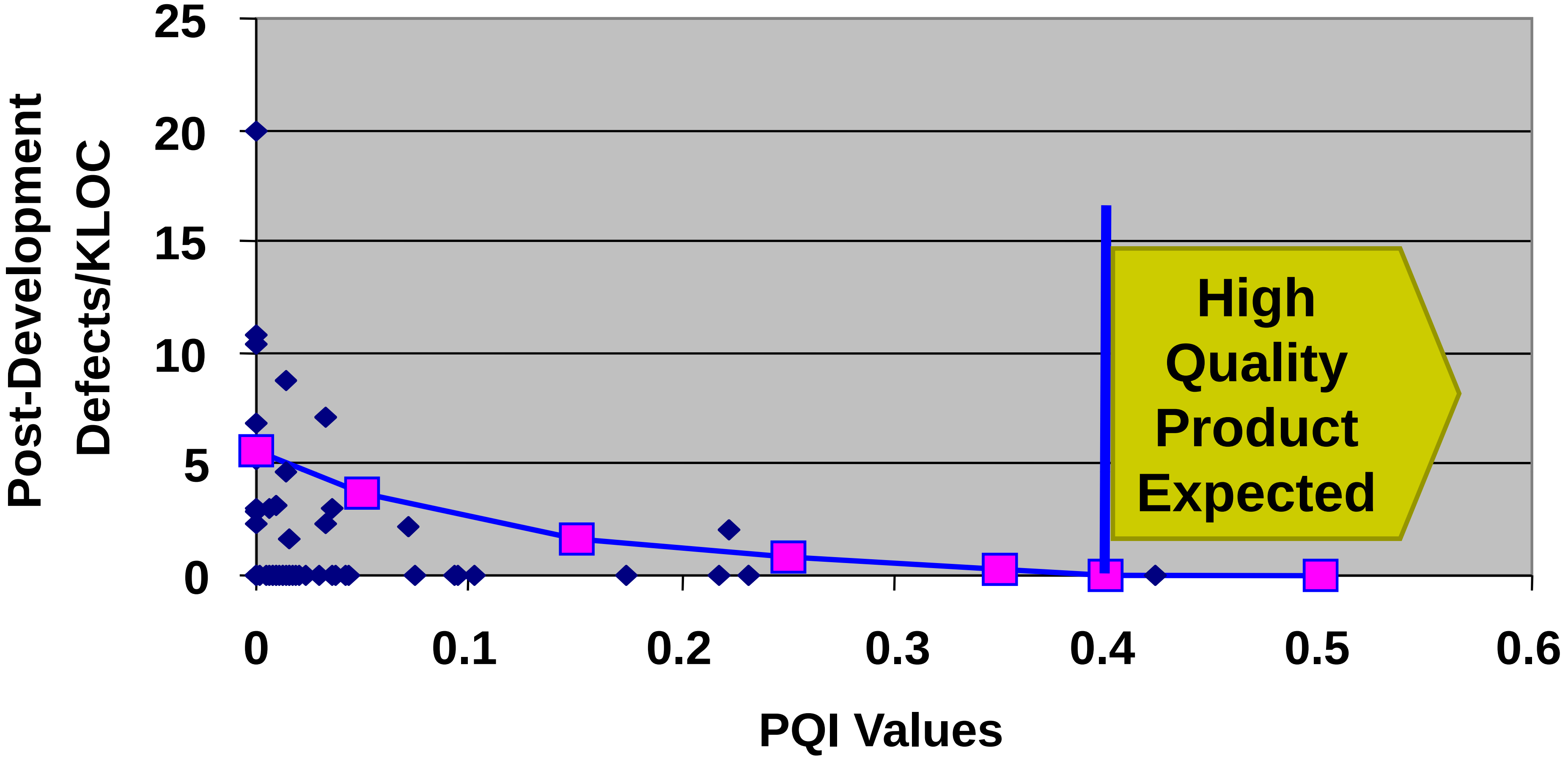
Test defects = 3



PQI = 0.04

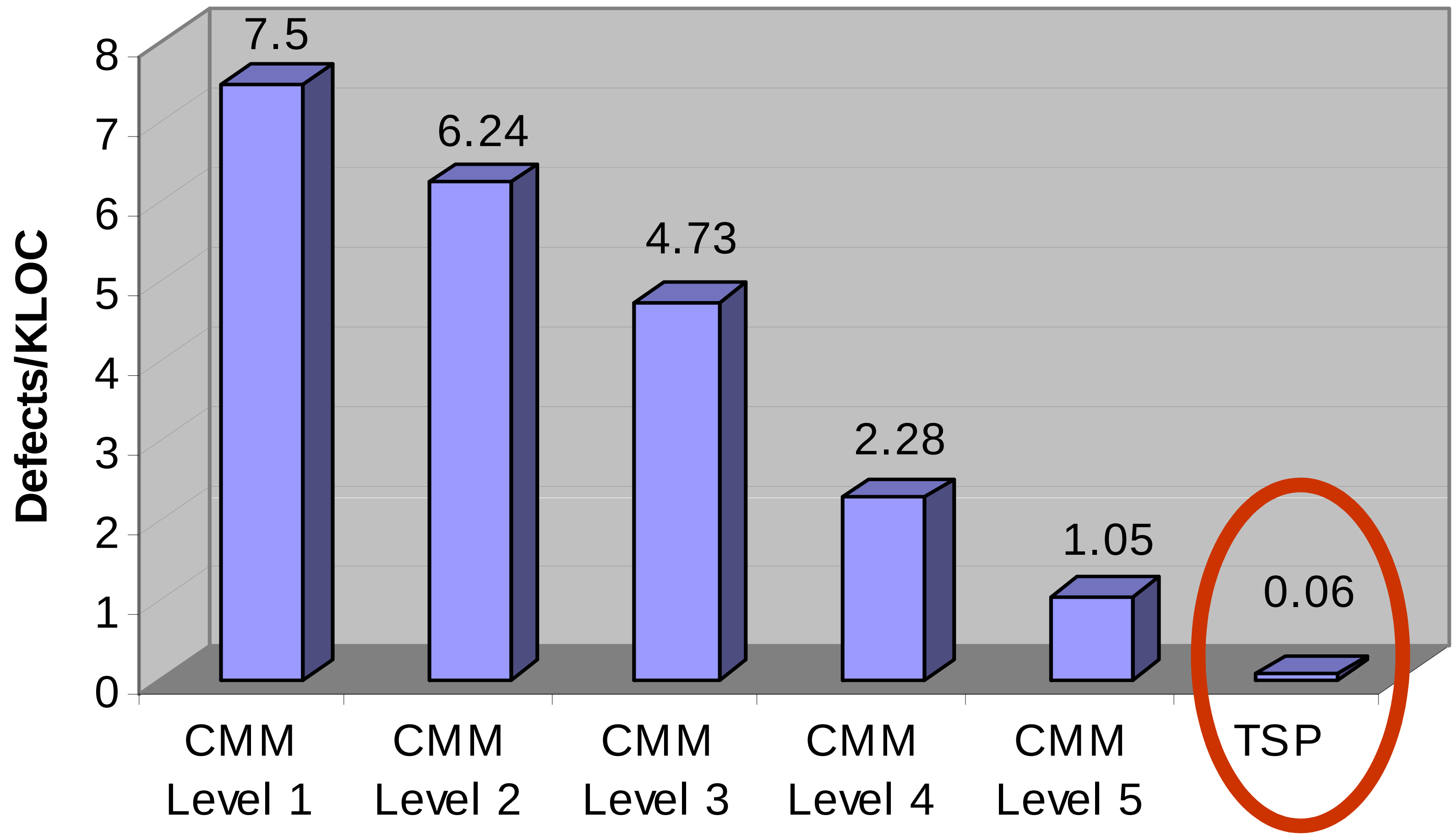
Reference: Watts Humphrey, PSP – A Self-Improvement Process for Software Engineers, Addison Wesley, 2005

PQI vs. Post-development Defects



Reference: Watts Humphrey, PSP – A Self-Improvement Process for Software Engineers, Addison Wesley, 2005

Defect Density of Delivered Software



Reference: Nooper Davis, Julia Mullany, SEI Technical Report 2003 - 014

Total Cost of Ownership (1/4):

Project Performance Study (see Ref.)

Project of

4 members,

About 4.2KLOC size.

Characterized by

Phase Yields (Y),

Phase Rates of Defect (R).

Yes – *TSP value followed*,

No – *lower yield*, or

higher inject. rate, or

lower removal rate

(*1) One member had a half of the removal rate of the others.

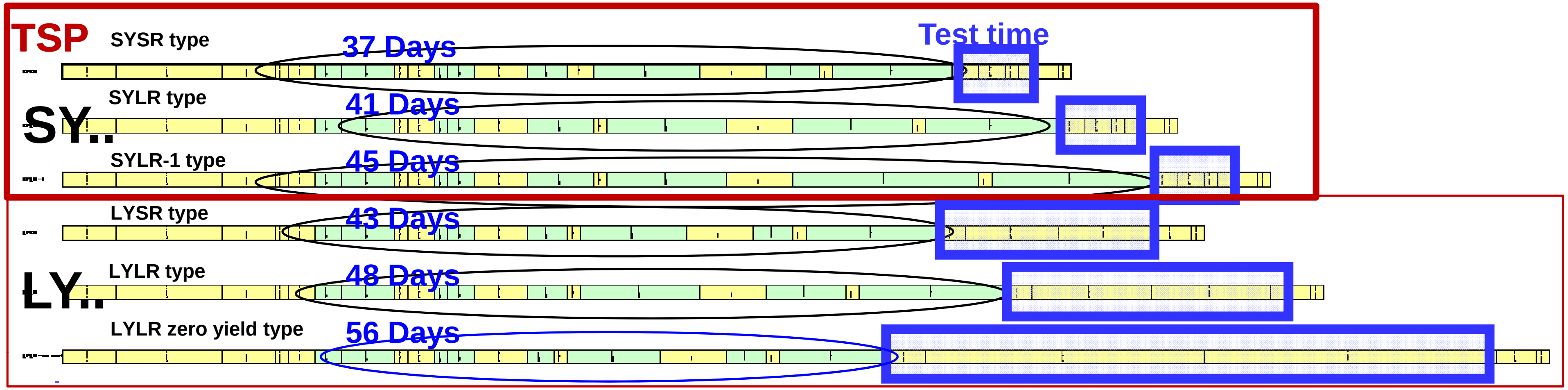
(*2) No yield achieved for Design and Code.

Project Type	***Description***	
Q- *Quality* **R** - *Rate* **S**- *Followed* **L** – *devidated*	**Focus on Phase Yields**	**Focus on Phase Rates**
SYSR	Yes	Yes
SYLR	Yes	No
SYLR-1**(*1)**	Yes	No
LYSR	No	Yes
LYLR	No	No
LYLR-zero yield **(*2)**	Zero yield	No

Ref.: Y.Akiyama, J. Over, Jim McHale, Anita Carton, Impact of Individual Performance to Organization, TSP Symposium 2006

Total Cost Ownership (2/4):
Project Management Tradeoff

- **Green segments** show review and inspection.
- **Blue box** show the time duration from the unit test through the system test.

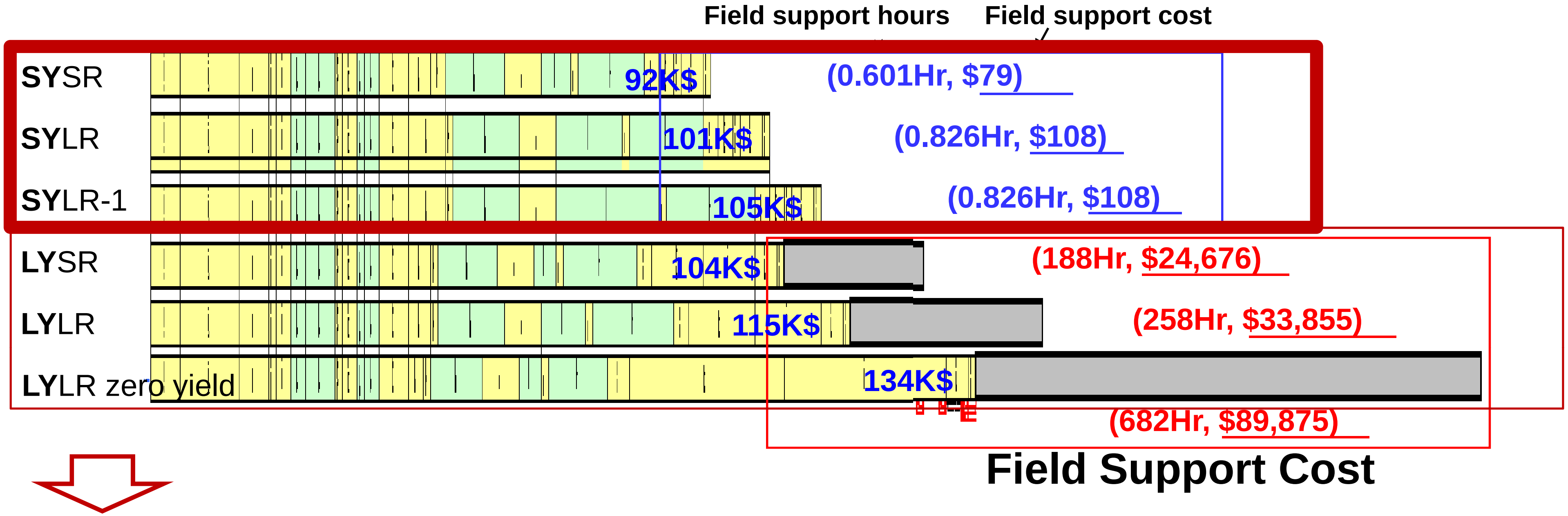


TSP saves project's testing time.

1) If the review & inspection time is longer, the test time becomes shorter.
2) Project length of the SY.. type is almost same or similar to that of LY types.

Total Cost Ownership(3/4):
Project's Field Support Cost

The grey boxes below show the cost needed to fix field defects.



The field support cost of

A) **SY*** type projects is negligible, i.e., very small.

B) **LY*** type projects is *not* negligible, i.e., not small.

Total Cost Ownership(4/4):
Field Cost

- **SY**xx type project - ***Solid profit***
  - Almost zero cost needs for the field support.
  - Most of the resources used for the project should be assigned to another project when completed.
- **LY**xx type project – ***Risky or may be Red profit***
  - 20 - 100% of the development cost must be planned for the field support.
  - Long tail maintenance must be expected.

Ref.: Y.Akiyama, J. Over, Jim McHale, Anita Carton, Impact of Individual Performance to Organization, TSP Symposium 2006

Further Remarks – 1
Communicating with other engineers

Process information is updated with experiences & knowledge:

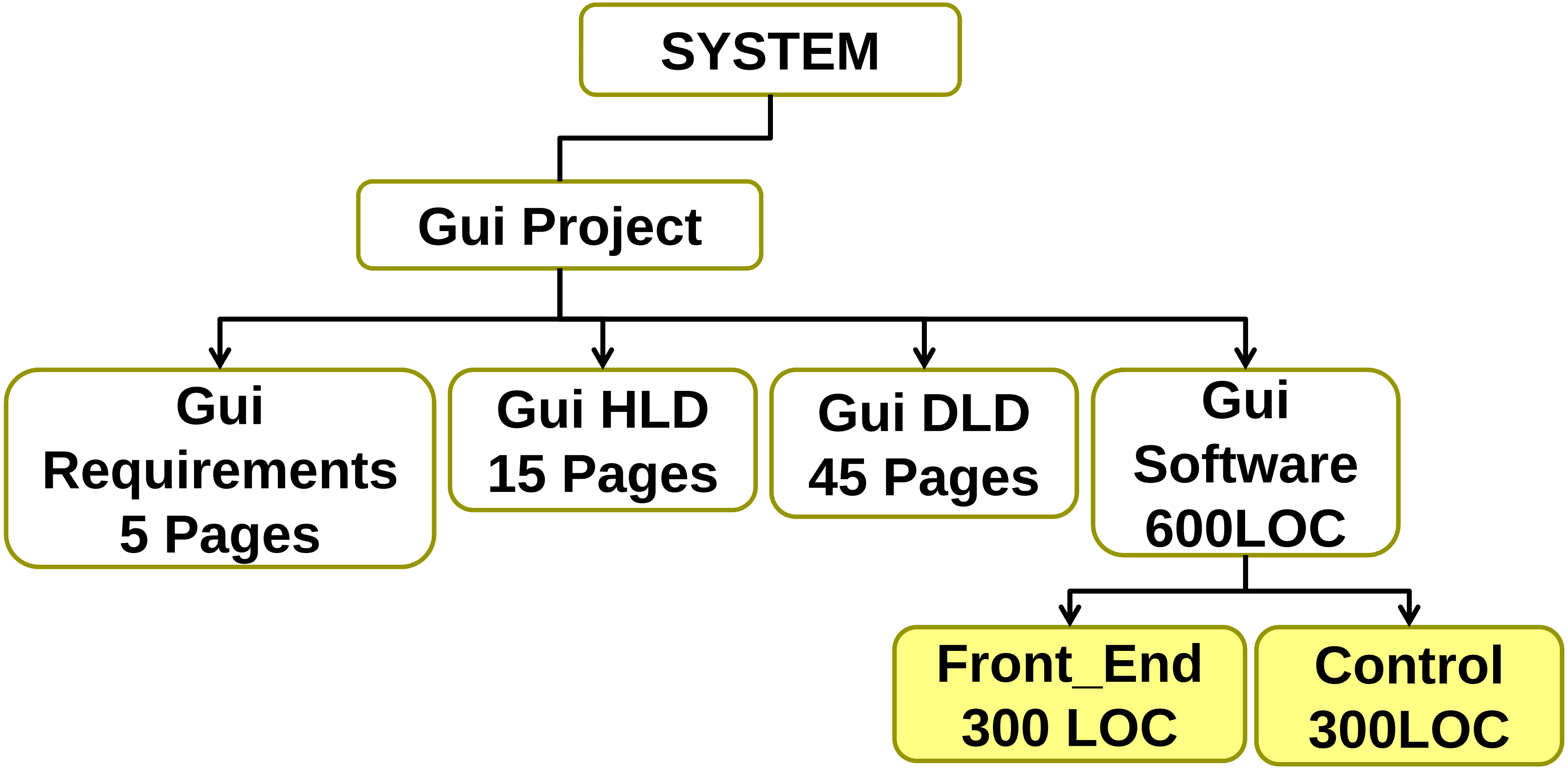
- Experiences and knowledge on requirement soliciting, design approach, implementation code, etc. are carried over to another project or another engineer
- Base data used for estimating and planning are continued and consistency is improved.

Further Remark - 2
Establishing Global Project Management

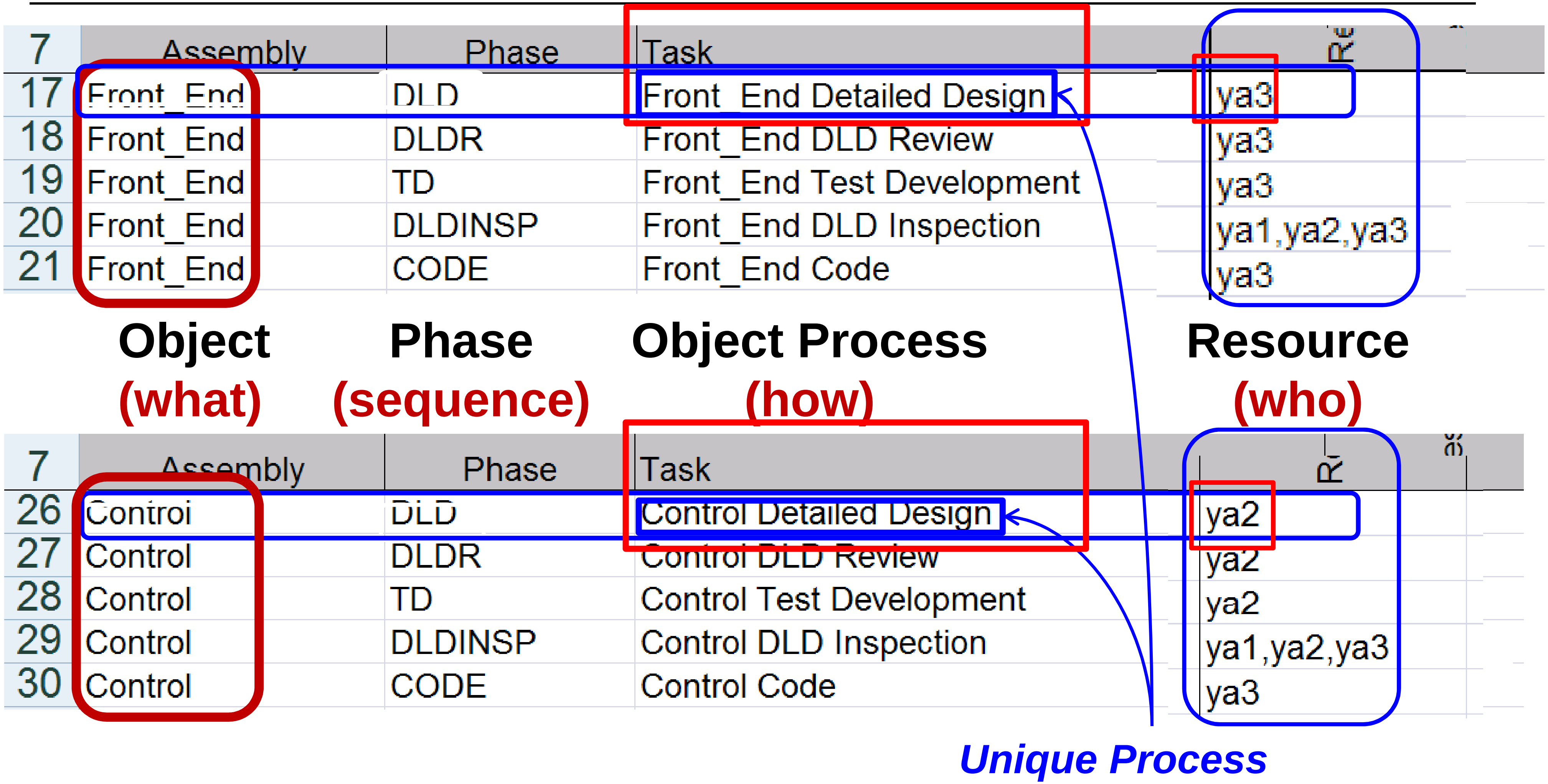
- *Active users* are living in active markets.
- *Such active markets* are located over the world.
- *Effective process tool support to realize the SOA based development* is necessary.

Here is a simple example by TSP.

TSP/SUMS defines a WBS for project.



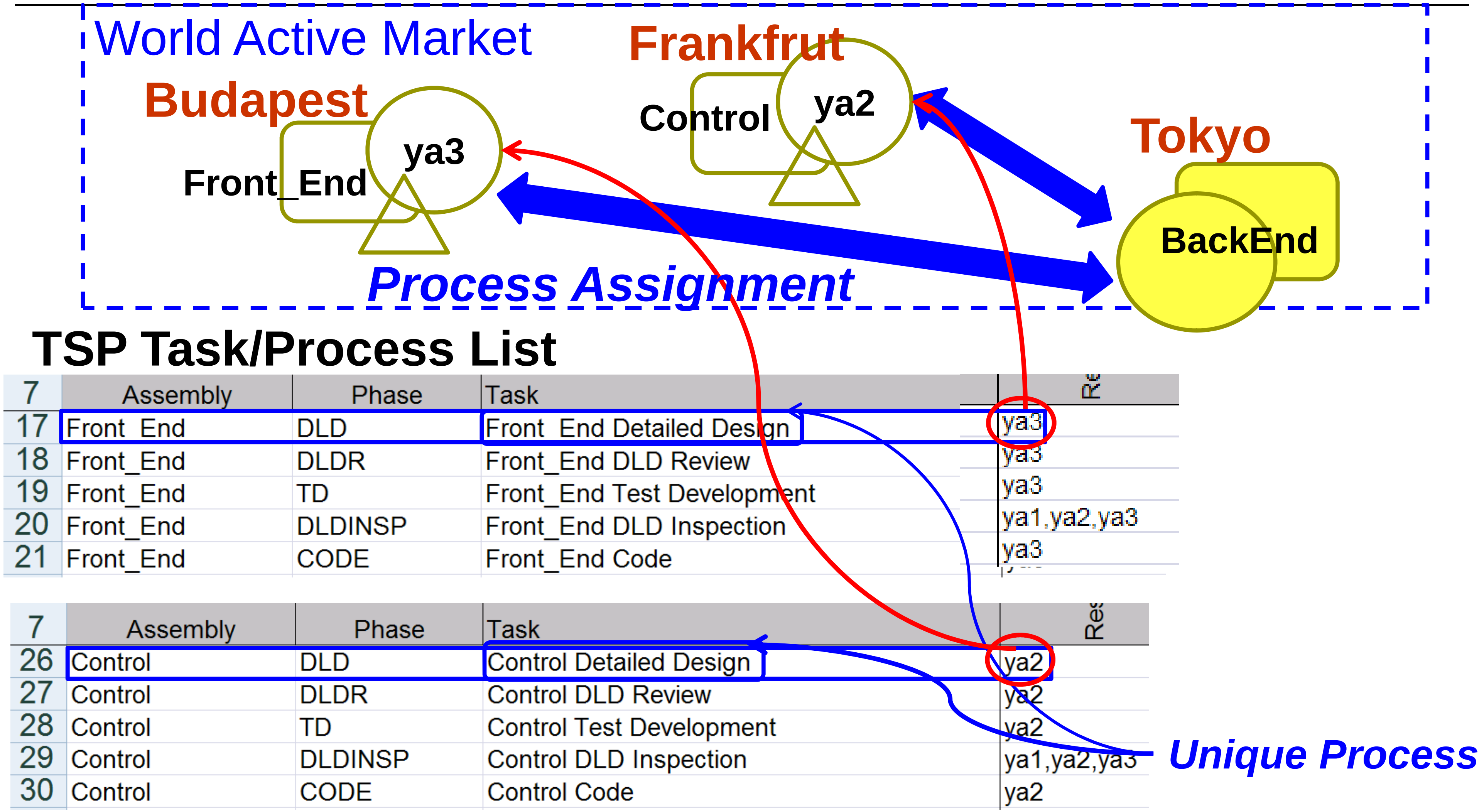
TSP generates TASK list (below is partial)



7	Assembly	Phase	Task	Re
17	Front_End	DLD	Front_End Detailed Design	ya3
18	Front_End	DLDR	Front_End DLD Review	ya3
19	Front_End	TD	Front_End Test Development	ya3
20	Front_End	DLDINSP	Front_End DLD Inspection	ya1,ya2,ya3
21	Front_End	CODE	Front_End Code	ya3

7	Assembly	Phase	Task	R
26	Control	DLD	Control Detailed Design	ya2
27	Control	DLDR	Control DLD Review	ya2
28	Control	TD	Control Test Development	ya2
29	Control	DLDINSP	Control DLD Inspection	ya1,ya2,ya3
30	Control	CODE	Control Code	ya3

TSP/TASK Global Assignments



TSP Task/Process List

7	Assembly	Phase	Task	Re
17	Front_End	DLD	Front_End Detailed Design	ya3
18	Front_End	DLDR	Front_End DLD Review	ya3
19	Front_End	TD	Front_End Test Development	ya3
20	Front_End	DLDINSP	Front_End DLD Inspection	ya1,ya2,ya3
21	Front_End	CODE	Front_End Code	ya3

7	Assembly	Phase	Task	Res
26	Control	DLD	Control Detailed Design	ya2
27	Control	DLDR	Control DLD Review	ya2
28	Control	TD	Control Test Development	ya2
29	Control	DLDINSP	Control DLD Inspection	ya1,ya2,ya3
30	Control	CODE	Control Code	ya2

Conclusion - 1

- A desired “proper process” is such as TSP/PSP and provides
  - Framework to manage individual activities,
  - Framework to manage team work.
- The PSP based training enables professional engineers who can show desirable high quality results as industry expects.
- The TSP establishes the effective team and realistic plan to be produced through the launch process.
- TSP process data are used to assess the plan accuracy and the quality of the project product before integration test or system test begins.
- The TSP can supports SOA based development.

Conclusion - 2

- *Good Process* transforms software engineer to professional who enriches its own process.
- *PSP instructors* and *TSP coaches* are effective supports for the transformation.
- Process transfers experiences and knowledge of software activities to other software activities and other engineers for better effectiveness and more efficiency.
- Every engineer uses process to communicate and negotiate on, and standardizes, and produces a new process to meet needs of your project and your organization.
- For SOA era, proper process is mandatory to become professional and to receive key inheritance.

Thank you for your attention,

and

Now for Q&A ….