Combinatorial Testing in Japan

- History, applications, techniques and tools -

Keizo TATSUMI

Fujitsu Ltd. / ASTER[1]

July 16, 2013

[1)]The author is a member of Specified Nonprofit Corporation "Association of Software Test Engineering (ASTER)" in Japan

Outline

- **Prevailing view to the history of CT**
- **The actual history**
- **Prehistory**
  - The early history of Fujitsu's mainframe computers
  - Test case design process in Fujitsu
- **The birth of CT**
  - Application of Design of Experiments
  - Test case design support system
- **Transferred CT technique to US**
- **Genealogy of combinatorial testing**
- **Current status of CT in Japan**
  - HAYST method
  - PictMaster
- **Concluding remarks**

Prevailing view to the history of CT

Mandl's paper is the first ??

Pairwise Testing

www.pairwise.org/default.html

Pairwise Testing

Combinatorial Test Case Generation

Pairwise (a.k.a. all-pairs) testing is an effective test case generation technique that is based on the observation that most faults are caused by interactions of at most two factors. Pairwise-generated test suites co two therefore are much smaller than exhaustive ones y finding defects.

$L_4(2^3)$ Orthogonal Array

Expt. No.	Column 1	2	3
1	1	1	1
2	1	2	2
3	2	1	2
4	2	2	1

What's New
Effectiveness of
Research Papers
Articles and Othe
Available Tools
About this Webs

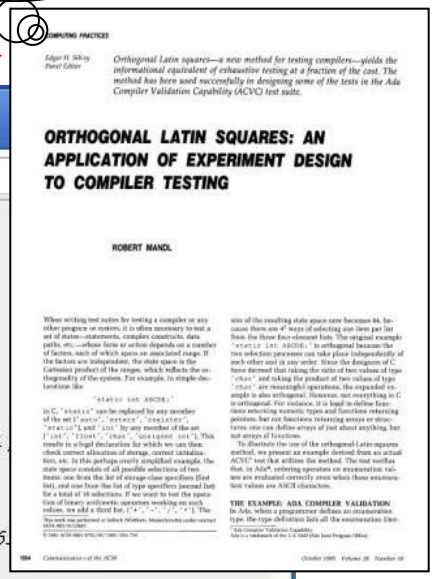
Maintained by Jacek Czerwonka Last updated: Ju

Pairwise Testing - Resear

www.pairwise.org/papers.asp

Research Papers

1. R. Mandl
   Orthogonal Latin squares: an application of experiment design to compiler testing
   Communications of the ACM, v.28 n.10, p.1054-1058, Oct. 1985
2. K. Tatsumi, S. Watanabe, Y. Takeuchi, H. Shimokawa
   Conceptual support for test case design
   Proc. 11th Intl. Computer Software and Applications Conf. (COMPSAC), October 1987,
3. K. Tatsumi
   Test Case Design Support System
   Proceedings of International Conference on Quality Control (ICQC), Tokyo, 1987, pp. 6
4. R. Brownlie, J. Prowse, and M.S. Phadke
   Robust Testing of AT&T PMX/StarMAIL using OATS
   AT&T Technical Journal, May/June 1992, pp. 41-47
5. K. Burroughs, A. Jain, and R. L. Erickson
   Improved quality of protocol testing through techniques of experimental design
   In Proceedings of the IEEE International Conference on Communications (Supercomm/ICC'94), May 1-5, New Orleans, Louisiana, USA. IEEE, May 1994, pp. 745-752
6. D. M. Cohen, S. R. Dalal, J. Parelius, G. C. Patton
   The Combinatorial Design Approach to Automatic Test Generation
   IEEE Software, Vol. 13, No. 5, pp. 83-87, September 1996
7. A.W. Williams, R.L. Probert
   A practical strategy for testing pair-wise coverage of network interfaces
   The Seventh International Symposium on Software Reliability Engineering (ISSRE '96) p. 246



ORTHOGONAL LATIN SQUARES: AN APPLICATION OF EXPERIMENT DESIGN TO COMPILER TESTING

ROBERT MANDL

The actual history

1950 1960 1970 1980 '85 1990 '95 2000 '05 2010

<Testing techniques>

(1957: Decision table)

~1967: Equivalence partitioning, Boundary value analysis

1970: Cause Effect Graph [Elmendorf]

Combinatorial techniques

Fujitsu

1984: Orthogonal array / Combination table [Satoh, Shimokawa]

1987: Test Case Design Support System [Tatsumi]

1985: Orthogonal Latin Squares [R. Mandl]

AT&T, Bellcore

(1989～)1992: OATS [Brownlie, Prowse, Phadke]

(1990～)1994: CATS [Sherwood]

(1992～)1994: AETG [Cohen, et. al]

(2000～)2004: PICT [Microsoft]

2000: CTE XL [Daimler Chrystler]

2007: FireEye 2009:ACTS (IPOG) [Lei, Kuhn]

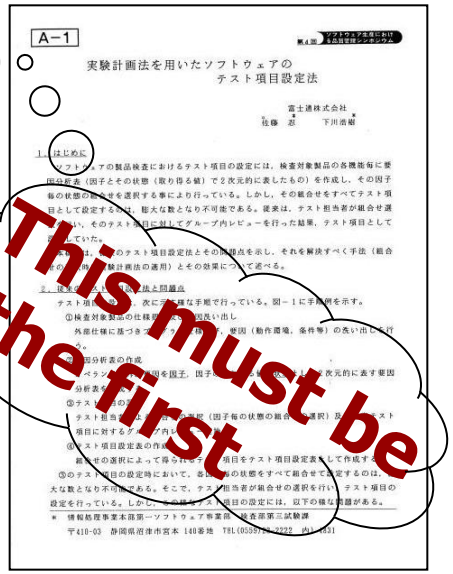
Input modeling techniques

(1976: Test Factor Analysis Method) [Fujitsu]

1988: Category-partition method [Ostrand, Balcer]

1993: Classification-tree method [Grochtmann, Grimm]

(*1)


A-1

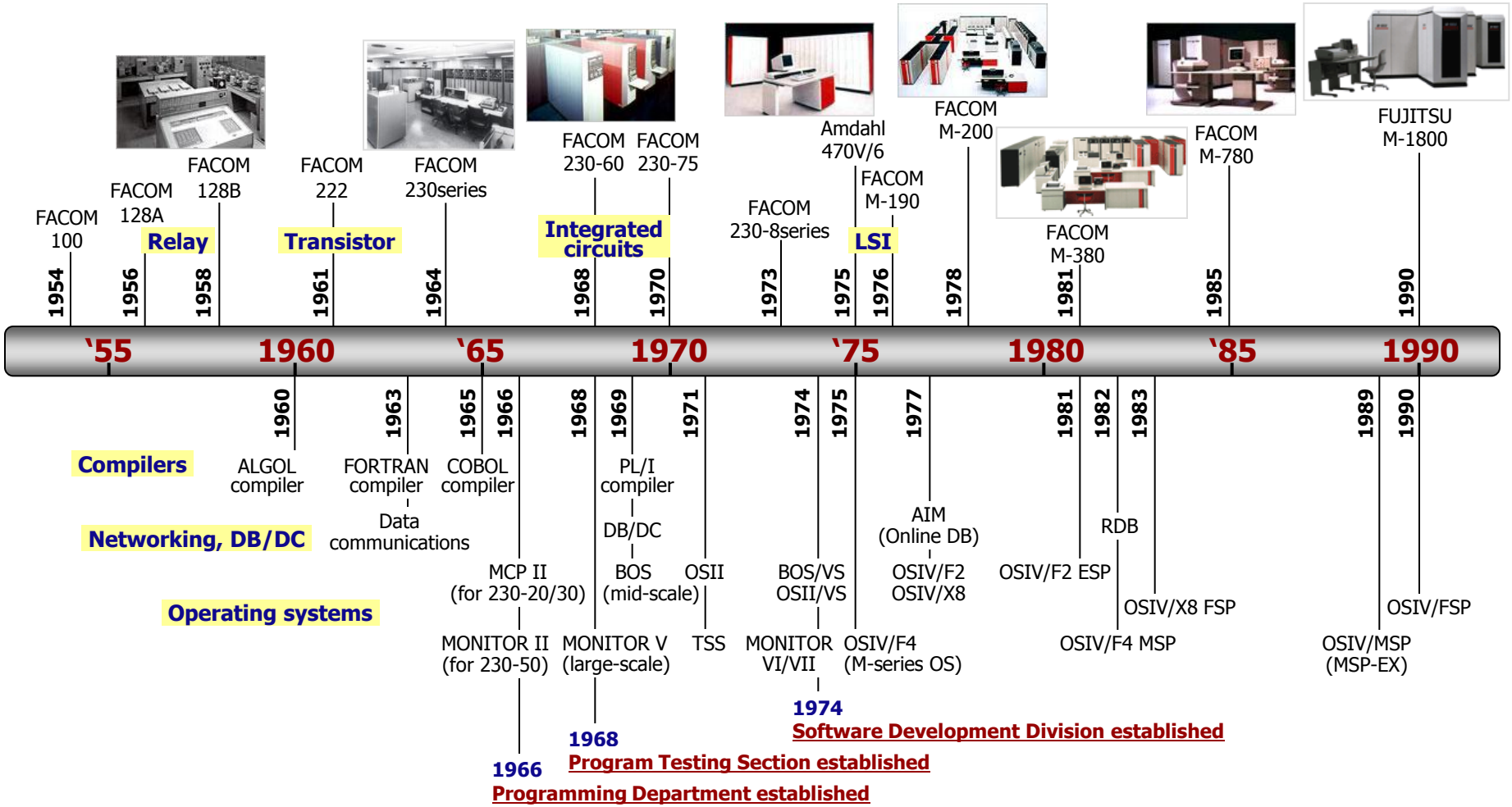
実験計画法を用いたソフトウェアのテスト項目設定法

This must be the first

(*1) S. Sato and H. Shimokawa, Methods for setting software test parameters using the design of experiments method (in Japanese), in Proceedings of 4th Symposium on Quality Control in Software, Union of Japanese Scientists and Engineers (JUSE), 1984, pp. 1–8.

Prehistory

⊙ The early history of Fujitsu's mainframe computers



M. A. Cusumano, "Japan's Software Factories: A Challenge to U.S. Management," Oxford University Press, 1991
K. Tatsumi, The History of Software Engineering and Software Testing - World and Japan - http://a-lifelong-tester.cocolog-nifty.com/Chronology/ChronologyEng.html

Prehistory

⊙ Test case design process in Fujitsu

(From early 1970s)

Step1. Test Classification

- Functions are divided into smaller units.

Step2. Test Factor Analysis

- Input conditions and environmental conditions (*Factors*) and their values (*States*) are identified from external specifications.

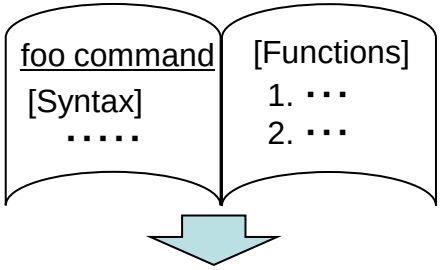
Step3. Test Case Generation

- Test cases are generated by combining the states of the factors.

Step4. Definition of Test Results

- Expected results are defined for each generated test cases.

<External specifications>



<Test factor table>

Factors		A	B	C	D
States	1	a1	b1	c1	d1
	2	a2	b2	c2	d2
	3	a3		c3	d3
	4			c4	

<Test case table>

Factors	A	B	C	D
Test case 1	a1	b1	c1	d1
Test case 2	a2	b2	c3	d2
...				
Test case n	a3	b1	c4	d3

The birth of CT (1983-1984)

⊙ Application of Design of Experiments

From Tatsumi's COMPSAC'87 presentation slides

STANDARD OF TEST CASE GENERATION

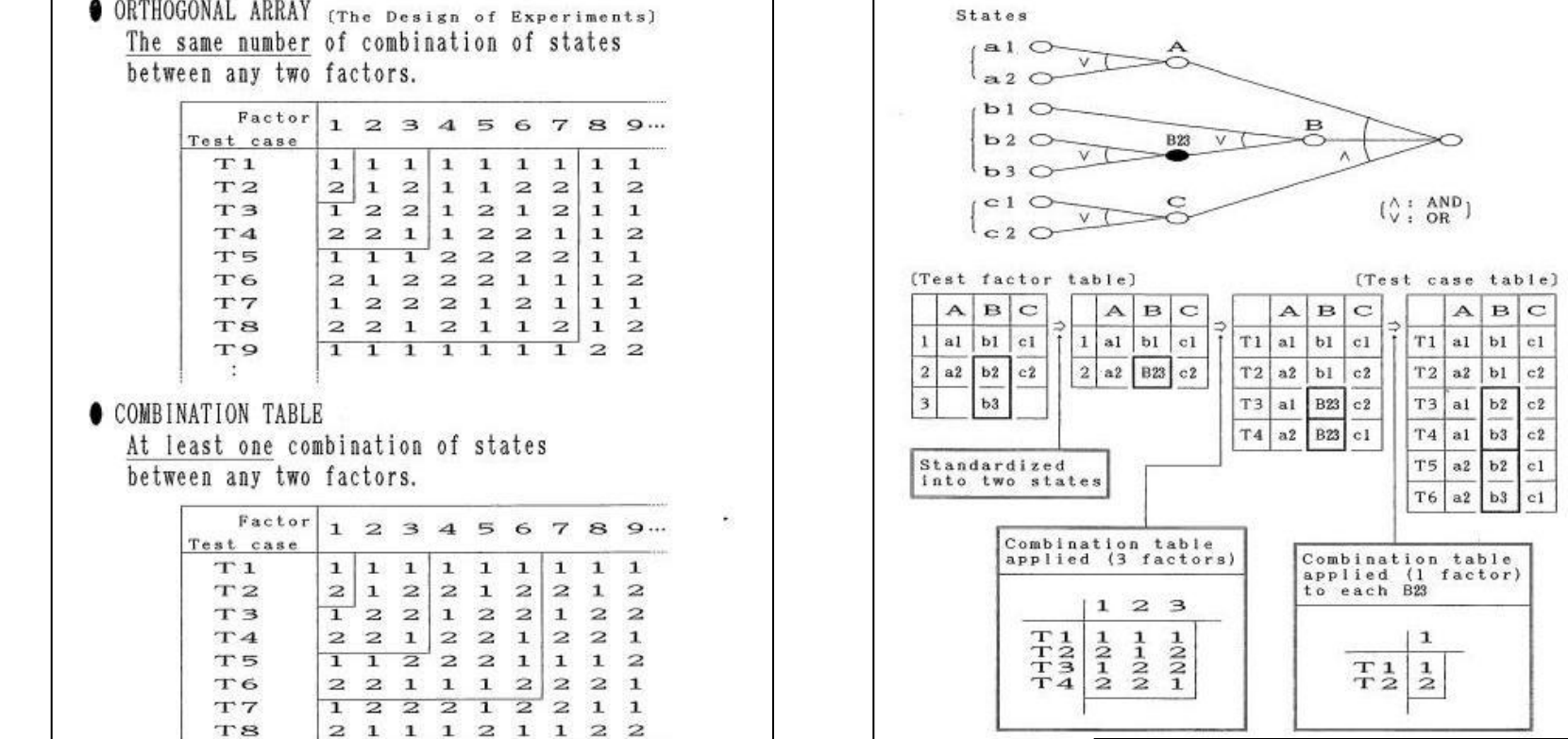
● ORTHOGONAL ARRAY (The Design of Experiments)
The same number of combination of states between any two factors.

Factor / Test case	1	2	3	4	5	6	7	8	9…
T1	1	1	1	1	1	1	1	1	1
T2	2	1	2	1	1	2	2	1	2
T3	1	2	2	1	2	1	2	1	1
T4	2	2	1	1	2	2	1	1	2
T5	1	1	1	2	2	2	2	1	1
T6	2	1	2	2	2	1	1	1	2
T7	1	2	2	2	1	2	1	1	1
T8	2	2	1	2	1	1	2	1	2
T9	1	1	1	1	1	1	1	2	2
:									

● COMBINATION TABLE
At least one combination of states between any two factors.

Factor / Test case	1	2	3	4	5	6	7	8	9…
T1	1	1	1	1	1	1	1	1	1
T2	2	1	2	2	1	2	2	1	2
T3	1	2	2	1	2	2	1	2	2
T4	2	2	1	2	2	1	2	2	1
T5	1	1	2	2	2	1	1	1	2
T6	2	2	1	1	1	2	2	2	1
T7	1	2	2	2	1	2	2	1	1
T8	2	1	1	1	2	1	1	2	2
T9	1	2	1	1	2	1	2	2	1
:									

APPLICATION OF THE COMBINATION TABLE



Constraints handling like Cause-Effect Graph
- Exclusive
- Inclusive
- One and only one

The birth of CT (1986-1987)

⊙ Test case design support system

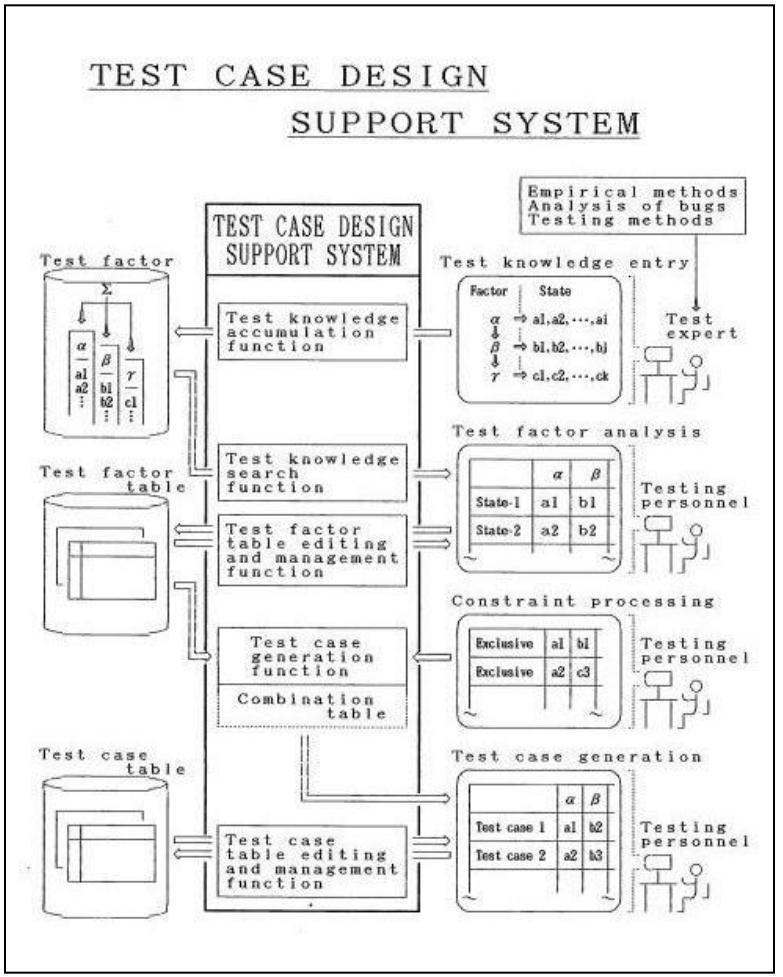
❖ Whole process support

- From test classification to test case generation
- Test documentation and revision management

❖ Refinement

- Generation algorithm
- Constraints handling
  - Between state-state, factor-state, factor-factor
    - » Exclusive combination
    - » No combination
    - » Required combination
    - » Grouping of states

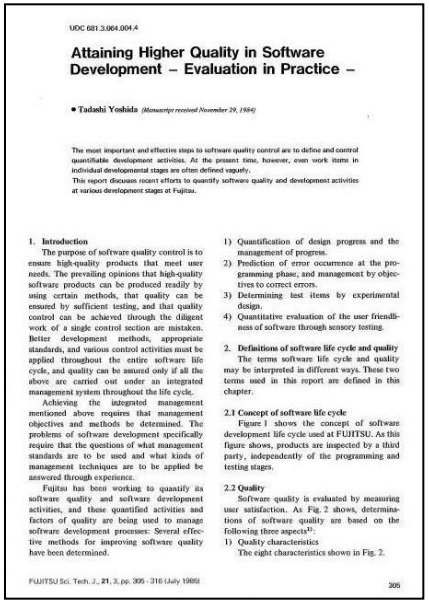
From Tatsumi's COMPSAC'87 presentation slides



Transferred CT technique to US

⊙ Publications in English

- T. Yoshida, “Attaining Higher Quality in Software Development - Evaluation in Practice”, FSTJ, 1985
- K. Tatsumi, "Conceptual Support for Test Case Design", COMPSAC'87
- K. Tatsumi, "Test Case Design Support System", ICQC'87



UDC 681.3.064.004.4

Attaining Higher Quality in Software Development – Evaluation in Practice –

• Tadashi Yoshida *(Manuscript received November 29, 1984)*

The most important and effective steps to software quality control are to define and control quantifiable development activities. At the present time, however, even work items in individual developmental stages are often defined vaguely.
This report discusses recent efforts to quantify software quality and development activities at various development stages at Fujitsu.

1. Introduction

The purpose of software quality control is to ensure high-quality products that meet user needs. The prevailing opinions that high-quality software products can be produced readily by using certain methods, that quality can be ensured by sufficient testing, and that quality control can be achieved through the diligent work of a single control section are mistaken. Better development methods, appropriate standards, and various control activities must be applied throughout the entire software life cycle, and quality can be assured only if all the above are carried out under an integrated management system throughout the life cycle.

Achieving the integrated management mentioned above requires that management objectives and methods be determined. The problems of software development specifically require that the questions of what management standards are to be used and what kinds of management techniques are to be applied be answered through experience.

Fujitsu has been working to quantify its software quality and software development activities, and those quantified activities and factors of quality are being used to manage software development processes: Several effective methods for improving software quality have been determined.

1) Quantification of design progress and the management of progress.
2) Prediction of error occurrence at the programming phase, and management by objectives to correct errors.
3) Determining test items by experimental design.
4) Quantitative evaluation of the user friendliness of software through sensory testing.

2. Definitions of software life cycle and quality

The terms software life cycle and quality may be interpreted in different ways. These two terms used in this report are defined in this chapter.

2.1 Concept of software life cycle

Figure 1 shows the concept of software development life cycle used at FUJITSU. As this figure shows, products are inspected by a third party, independently of the programming and testing stages.

2.2 Quality

Software quality is evaluated by measuring user satisfaction. As Fig. 2 shows, determinations of software quality are based on the following three aspects[2]:

1) Quality characteristics

The eight characteristics shown in Fig. 2.

FUJITSU Sci. Tech. J., 21, 3, pp. 305-316 (July 1985)

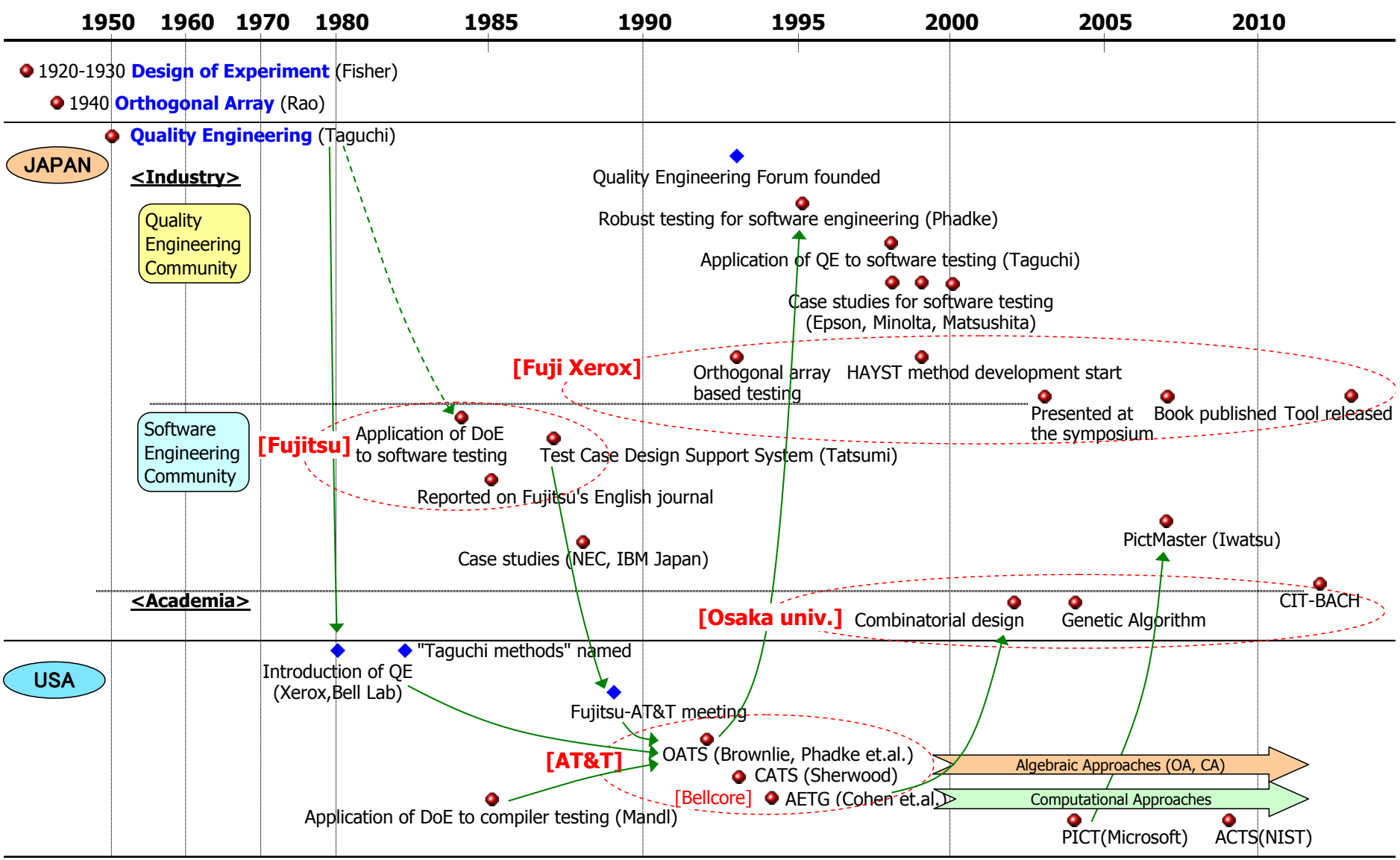
⊙ Introducing to AT&T in 1989

(According to Yoshida's book)

- Yoshida was requested for explanation about application of DoE from some AT&T engineer who had read his FSTJ paper.
- He and colleagues had a meeting with AT&T engineers and explained the technique at Marriott hotel in Somerset, N. J. on 30th April in 1989. It was while his travelling as a member of the 1st Software Production Quality Control Study Team of JUSE.
- Tatsumi’s ICQC paper was the one of documents for explanation.

T. Yoshida, "Attaining Higher Quality in Software Development - Evaluation in Practice -", Fujitsu Scientific and Technical Journal, Vol.21, pp.305-316, 1985
T. Yoshida, “Technology transmission and transfer (in Japanese)", pp.113-114, JUSE press, 1994

Genealogy of combinatorial testing



Current status of CT in Japan (1/2)

⊙ Software testing community

- ❖ **ASTER**: Association of Software Test EngineeRing
  - Non profit organization for promoting software testing
  - Operated / Managed by volunteers of software testing experts
  - Activities
    - **Conference** - JaSST: Japan Symposium on Software Testing
    - **Certification** - JSTQB: Japan Software Testing Qualifications Board, member of ISTQB
    - **International collaboration** - ASTA (Asian Alliance) / ISO-IEC (International Std.)
    - **Education** - Seminars / Open materials / Univ. Curriculum / Test.SSF (Skills Standard)
    - **Research** - Research groups on Test Architecture and on Bug Analysis

Current status of CT in Japan (2/2)

⊙ Number of papers and presentations on CT

1984-'90: **9**, '91-'95: **0**, '96-'00: **6**, '01-'05: **9**, '06- : **31**

⊙ Applications of CT

(Questionnaire survey to the software testing experts of 5 major IT vendors)

- ❖ CT has already been applied in all the vendors
- ❖ Tools
  - PICT, PictMaster, ACTS, HAYST, In-house tools
- ❖ Issues
  - Test design phase before combination is essential
  - Selection criteria for CT with other similar testing techniques

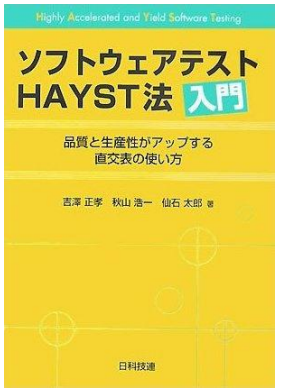
⊙ Popular tools

- ❖ HAYST method
- ❖ PictMaster

HAYST method (1/4)

About

- Orthogonal array based testing technique
- Developed by Fuji Xerox in 2000
- "Introduction to the HAYST method" in 2007



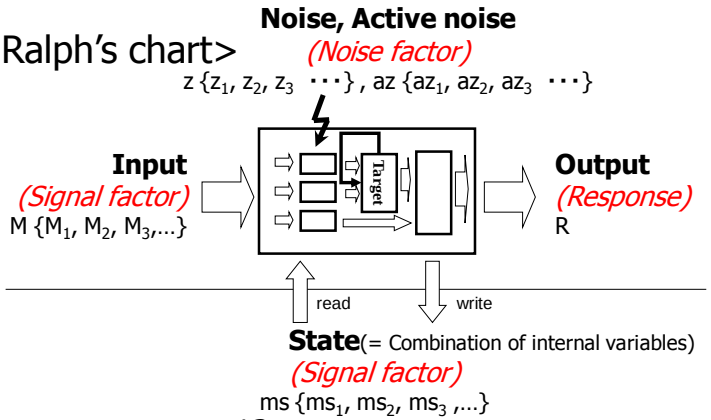
Test case design methodology

- "Function Verification table" for test analysis
- "Ralph's chart" for modeling the system under test
- "Function Level table" for extracting factors and levels

<FV table>

No.	Objective function	Verification contents	Testing technique

<Ralph's chart>



<FL table>

No.	Factor	Level 1	Level 2	...	Level N

HAYST method (2/4)

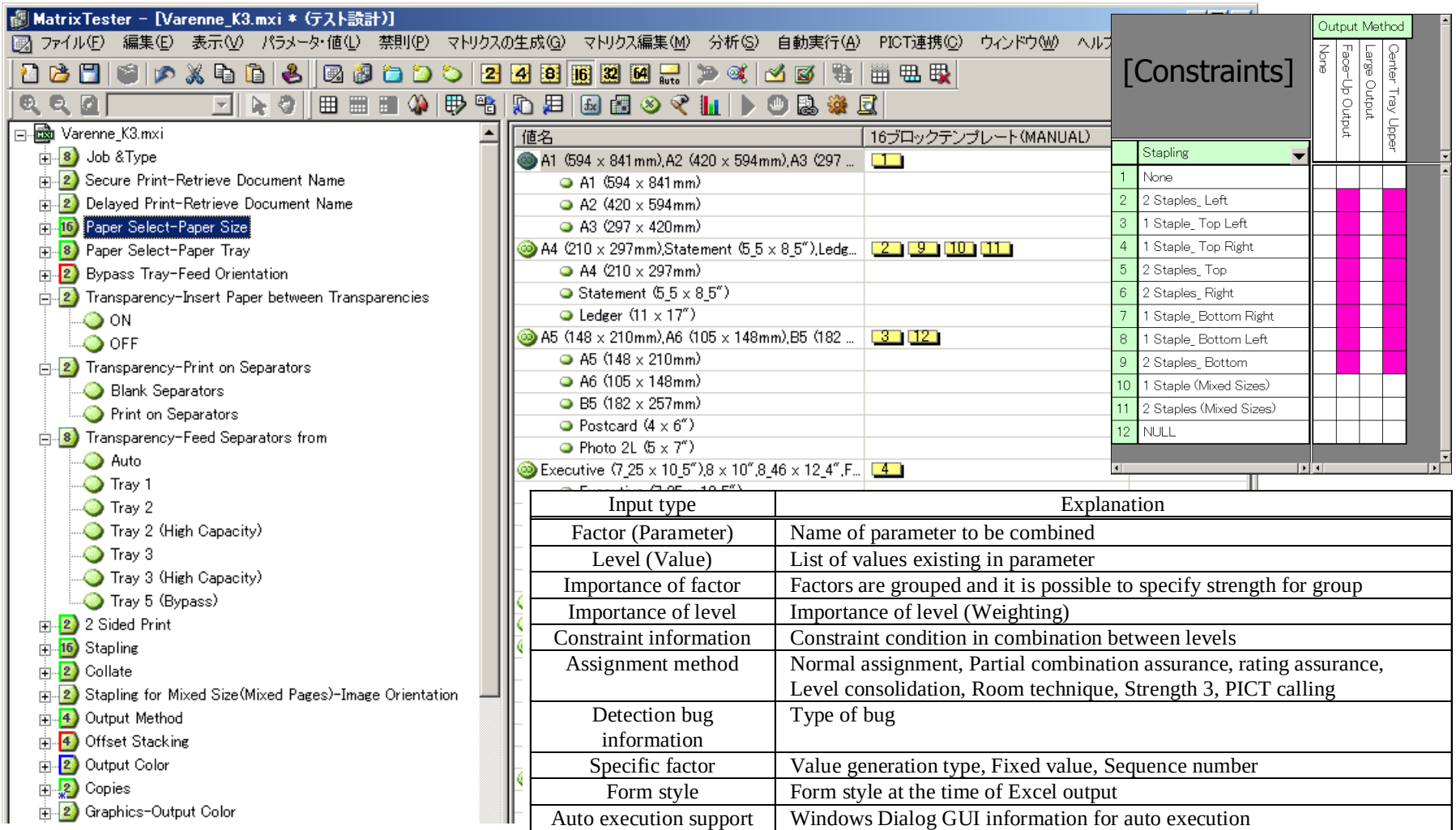
- **HAYST method tool "MatrixTester"**
  - Transform from OA (strength 2) templates L_4 - L_{256}
  - Constraints handling
  - Partial combination assurance (3, 4, 5 way)
  - Combinatorial coverage calculation
- **Applications**
  - MatrixTester
    - 560 licenses (in-house) and 23 companies (160 licenses)
      - Manufacturers (Automobile, Medical), Financials, Retailing, Consumer Electronics, Communications Equipment, Chemicals
  - Introductory book
    - 4000 copies

K. Akiyama et al., "Development and Evaluation of HAYST Method Tool," in Proc. SoMeT, 2010, pp.398-414.
http://www.hayst.com/Documents/HAYST-MethodTool.pdf (slides) http://www.hayst.com/Documents/20101001-SoMeT.pdf

HAYST method (3/4)

(Source: Koichi Akiyama, Fuji Xerox)

⊙ MatrixTester : input (factors, levels, constraints)

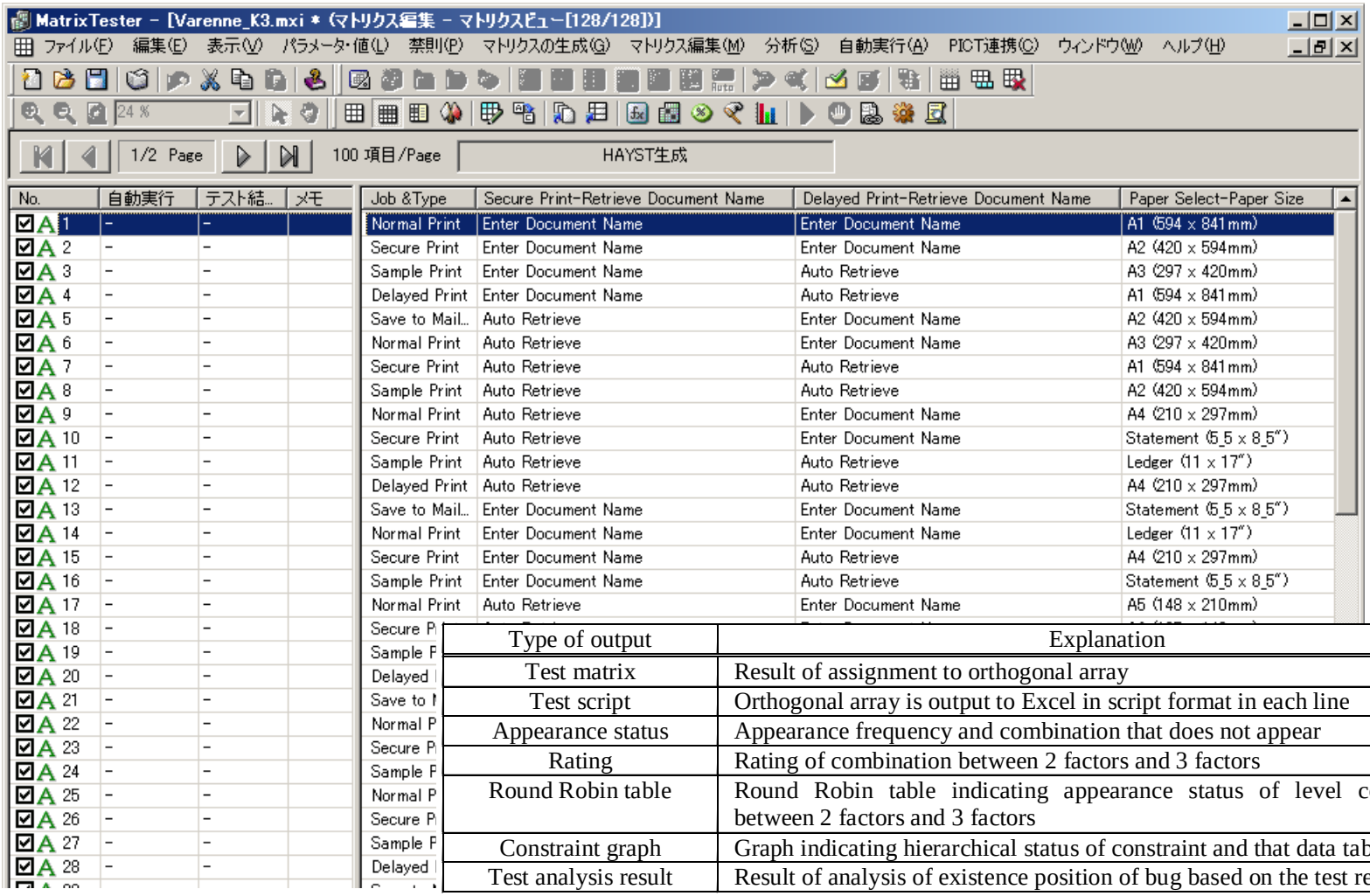


Input type	Explanation
Factor (Parameter)	Name of parameter to be combined
Level (Value)	List of values existing in parameter
Importance of factor	Factors are grouped and it is possible to specify strength for group
Importance of level	Importance of level (Weighting)
Constraint information	Constraint condition in combination between levels
Assignment method	Normal assignment, Partial combination assurance, rating assurance, Level consolidation, Room technique, Strength 3, PICT calling
Detection bug information	Type of bug
Specific factor	Value generation type, Fixed value, Sequence number
Form style	Form style at the time of Excel output
Auto execution support information	Windows Dialog GUI information for auto execution

HAYST method (4/4)

(Source: Koichi Akiyama, Fuji Xerox)

⊙ MatrixTester : output (Test matrix)



No.	自動実行	テスト結...	メモ
1	-	-	
2	-	-	
3	-	-	
4	-	-	
5	-	-	
6	-	-	
7	-	-	
8	-	-	
9	-	-	
10	-	-	
11	-	-	
12	-	-	
13	-	-	
14	-	-	
15	-	-	
16	-	-	
17	-	-	
18	-	-	
19	-	-	
20	-	-	
21	-	-	
22	-	-	
23	-	-	
24	-	-	
25	-	-	
26	-	-	
27	-	-	
28	-	-	

Job &Type	Secure Print-Retrieve Document Name	Delayed Print-Retrieve Document Name	Paper Select-Paper Size
Normal Print	Enter Document Name	Enter Document Name	A1 (594 x 841mm)
Secure Print	Enter Document Name	Enter Document Name	A2 (420 x 594mm)
Sample Print	Enter Document Name	Auto Retrieve	A3 (297 x 420mm)
Delayed Print	Enter Document Name	Auto Retrieve	A1 (594 x 841mm)
Save to Mail...	Auto Retrieve	Enter Document Name	A2 (420 x 594mm)
Normal Print	Auto Retrieve	Enter Document Name	A3 (297 x 420mm)
Secure Print	Auto Retrieve	Auto Retrieve	A1 (594 x 841mm)
Sample Print	Auto Retrieve	Auto Retrieve	A2 (420 x 594mm)
Normal Print	Auto Retrieve	Enter Document Name	A4 (210 x 297mm)
Secure Print	Auto Retrieve	Enter Document Name	Statement (5.5 x 8.5")
Sample Print	Auto Retrieve	Auto Retrieve	Ledger (11 x 17")
Delayed Print	Auto Retrieve	Auto Retrieve	A4 (210 x 297mm)
Save to Mail...	Enter Document Name	Enter Document Name	Statement (5.5 x 8.5")
Normal Print	Enter Document Name	Enter Document Name	Ledger (11 x 17")
Secure Print	Enter Document Name	Auto Retrieve	A4 (210 x 297mm)
Sample Print	Enter Document Name	Auto Retrieve	Statement (5.5 x 8.5")
Normal Print	Auto Retrieve	Enter Document Name	A5 (148 x 210mm)

Type of output	Explanation
Test matrix	Result of assignment to orthogonal array
Test script	Orthogonal array is output to Excel in script format in each line
Appearance status	Appearance frequency and combination that does not appear
Rating	Rating of combination between 2 factors and 3 factors
Round Robin table	Round Robin table indicating appearance status of level combination between 2 factors and 3 factors
Constraint graph	Graph indicating hierarchical status of constraint and that data table
Test analysis result	Result of analysis of existence position of bug based on the test result

PictMaster (1/2)

- **About**
  - Pairwise testing tool with Excel based GUI
  - Calls PICT as a test case generation engine
  - Developed by Toshiro Tsurumaki of Iwatsu in 2008
- **Applications**
  - 20,000 downloads (total) from 2008
  - System testing for Business phone system in IWATSU
  - Automotive OS testing
  - Large scale package software testing
- **Download at**

  http://en.sourceforge.jp/projects/pictmaster/releases/

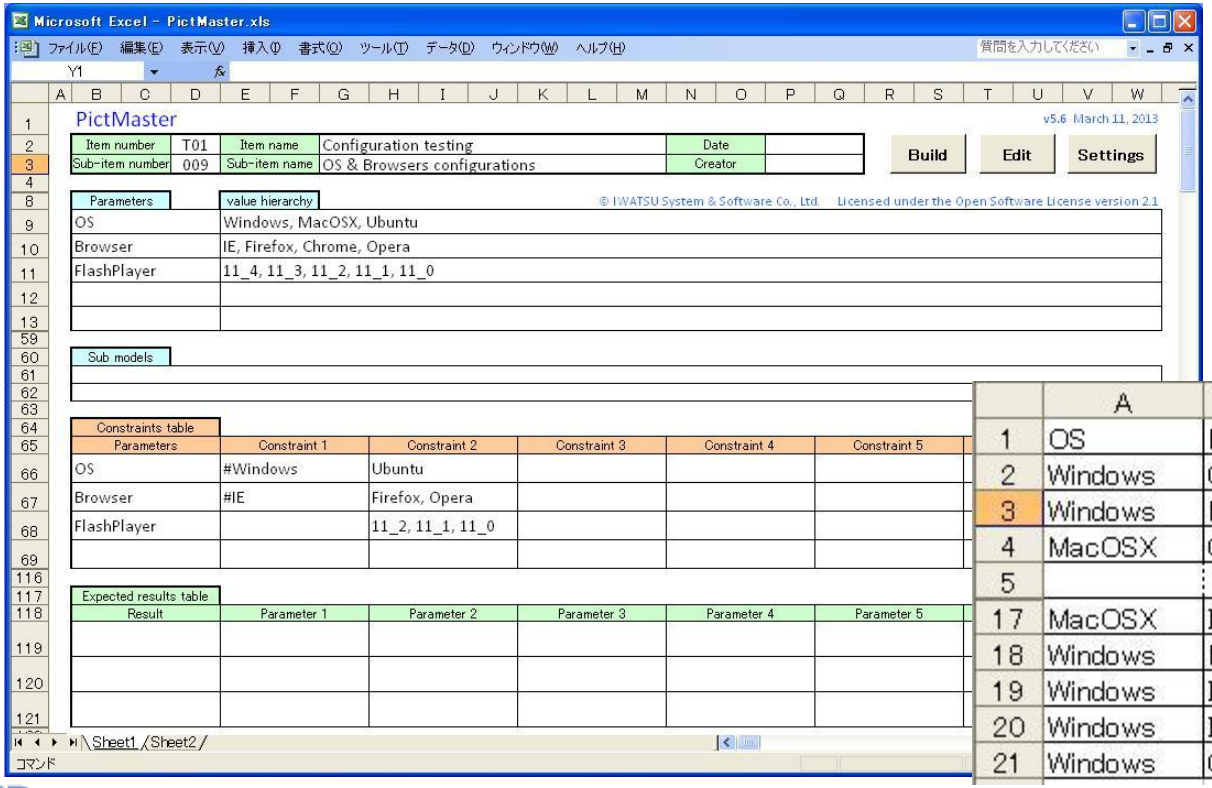
Mr. Tsurumaki can be contacted at LinkedIn "Combinatorial and Pairwise Testing" group.
http://www.linkedin.com/groups/Combinatorial-Pairwise-Testing-4243185

PictMaster (2/2)

(Source: Toshiro Tsurumaki, IWATSU SYSTEM & SOFTWARE)

⊙ Major Features

- ❖ GUI support
- ❖ Constraints table (for generating PICT constraints)
- ❖ Expected results table



	A	B	C
1	OS	Browser	FlashPlayer
2	Windows	Chrome	11_0
3	Windows	Firefox	11_3
4	MacOSX	Chrome	11_1
5			
17	MacOSX	IE	11_4
18	Windows	Firefox	11_0
19	Windows	IE	11_1
20	Windows	IE	11_3
21	Windows	Chrome	11_2

Concluding remarks

- **Lessons learned from the history of CT**

  For evolution of software testing technology, it needs

  - Openness and collaboration
  - Tools and methodologies

- **Challenges in Japan**

  - More focus on upper phase of test process
    - Test architecture design, input modeling
  - Guidelines for choosing the right technique from among many testing techniques
    - For what type of software should CT be applied?