

SPLC 2023 Keynote

The 20-year journey of SPLE in Hitachi and the next

August 31st, 2023

Hideto OgawaDistinguished Researcher, Ph.D.Kentaro YoshimuraChief Researcher, Ph.D.

Research and Development group Hitachi, Ltd.

© Hitachi, Ltd. 2023. All rights reserved.

HITACHI Inspire the Next

Content

Overview (Dr. Ogawa)
 Technical Journey (Dr. Yoshimura)
 The next (Dr. Ogawa)

© Hitachi, Ltd. 2023. All rights reserved.



1. Overview

Self-Introduction



Hideto OGAWA

Distinguished Researcher Service System Innovation Center Research & Development Group Hitachi, Ltd.

PhD (Information Science). He joined Hitachi in 1996 and involved to software engineering research, such as software testing, formal method, quality assurance and machine learning.

Information Processing Society of Japan, Director The Institute of Electronics, Information and Communication Engineers, Member Japan Society of Software Science and Technology, Member Shizuoka University, Visiting Professor Japan Advanced Institute of Science and Technology, Industryacademia-government collaboration Visiting Professor





SPLC'17 in Sevilla Hitachi young best paper award

3



Hitachi's Social Innovation Business



Business that leverages Hitachi's IT, OT and products to resolve social challenges together with customers

- Work with customers to support people's quality of life and fosters a sustainable society through the use of data and technology
- Expand the Social Innovation Business in diverse industries using three global trends of Digital, Green, and Connective, as transformation drivers

Digital transforming social systems

finance, public services, electric power, transportation, communications, etc.

Green transforming social infrastructures

power grids, nuclear power, railways, etc.

Connective

transforming industrial systems

factories, tech, buildings, homes, etc.

• Globally expand Lumada as customer co-creation framework of the Social Innovation Business



Software Development Improvement



SPLE

Model-based Development

Testing

Verification

Why SPLE? Too much Variations



Having many models was a competitive advantage.

Kato, Kawakami, Myojin, Ogawa, Hirono, Hasegawa: Case Study of Applying SPLE to Development of Network Switch Products, Proc. of SPLC2013, pp.198-207(2013.08) © Hitachi, Ltd. 2023. All rights reserved.

8

Problems

The existence of many software variants has had a tremendous negative impact on software development

- About Implementation :
 - if a common bug is found, duplicate modification operations are needed
- About Testing:
 - if the tester is changed, new tester need to rebuild the test assets.
- About Documentation:
 - Documents which has similar contents are created by different organization

Kato, Kawakami, Myojin, Ogawa, Hirono, Hasegawa: Case Study of Applying SPLE to Development of Network Switch Products, Proc. of SPLC2013, pp.198-207(2013.08) © Hitachi, Ltd. 2023. All rights reserved.

For Developers

ForUsers

Example of solutions

HITACHI Inspire the Next

Source code integration & Guidelines for dev. and management

Kato, Kawakami, Myojin, Ogawa, Hirono, Hasegawa: Case Study of Applying SPLE to Development of Network Switch Products, Proc. of SPLC2013, pp.198-207(2013.08) © Hitachi, Ltd. 2023. All rights reserved

10

System for sharing technical knowledge within the Hitachi Group

Research and Development Group Corporate management Division Engineering Support Division

Special Interest Group on Software Development Improvement

Business Units in Hitachi and Hitachi Group Companies

HITACHI Inspire the Next

	2000	2010	2015	2020	
Applications	E E	MSOFT'06 (Yosł	nimura)	S	PLC'23 (Yoshimura)
			6 (<i>lida</i> ; Yoshimura	*)	
		S	SPLC'13 (Kato)		
			SPLC'14 (Koda	ama)	
Organizational activ	ities	SPL	.C'11 (Yoshimura)		
Methodoloav	S	SPLC'06 (Yoshim	ura)		
		MSR'08 (Yosh	nimura)		
		SPLC'10 (Yoshimura)		
		Р	LEASE'12 (Shimba	ra)	
			SPLC'13 (Cleme	ens; Yoshimura*)	
			SPLC'13 (Tsuch	<i>iya;</i> Kato*)	
			SPLC'15	ō (<i>Vasilevskiy;</i> Shii	mbara*)
Collaborative Research		◆ 大阪大学 OSAKA UNIVERSITY ◆	早稲田大学 WASEDA University	TEF	* co-author

2. Technical Journey

Self Introduction

Kentaro YOSHIMURA

Chief Researcher Autonomous Control Research Department Connective Automation Innovation Center Research & Development Group Hitachi, Ltd.

PhD (Information Science and Technology). He joined Hitachi in 2001 and involved to software engineering research of control systems, such as automotive, robotics and industrial systems.

IEEE, Senior Member Information Processing Society of Japan, Member Japan Society of Mechanical Engineering, Member APSEC2023, Co-Chair (APSEC: Asia Pacific Software Engineering Conference)

Current Status of SPLE – A View from Industry

SPLE is recognized an established software development approach

International Standard

IEC.	We Interna	bstc tional El	DICE ectrotechnical Commission
HOME	SIGN IN	HELP	CART (0)

ISO/IEC 26550:2015

Software and systems engineering - Reference model for product line engineering and management

ISO/IEC JTC 1/SC 7 | Additional information

Abstract

PREVIEW

ISO/IEC 26550:2015 is the entry point of the whole suite of International Standards for software and systems product line engineering and management.

The scope of this International Standard is to

- provide the terms and definitions specific to software and systems product line engineering and management,

 define a reference model for the overall structure and processes of software and systems product line engineering and management and describe how the components of the product line reference model fit together, and Show more »

Syllabus for Fundamental Information Technology Engineer Examination (Japan)

大分類4:開発技術 中分類13:ソフトウェア開発管理技術

<u>開発プロセス・手法</u>

【目標】

- ▶ ソフトウェア開発プロセスに関する代表的な手法の考え方を理解し、担当する事項に 適用する。
- アジャイルの概要、アジャイルソフトウェア開発手法の考え方を理解し、担当する事項 に適用する。

(1) ソフトウェア開発手法

ソフトウェア開発モデル

ソフトウェア開発の効率化や品質向上のために用いられるソフトウェア開発モデルの考 え方,必要性,代表的なソフトウェア開発モデルの考え方を理解する。

用語例 ウォーターフォールモデル、プロトタイピングモデル、アジャイル、ソフトウェアプロダクトライン、段階的モデル(Incremental Model)、進展的モデル (Evolutionary Model)

(1) Software development approaches e.g. water fall, agile, **software product line**…

https://www.ipa.go.jp/shiken/syllabus/t6hhco000000ijhj-att/syllabus_fe_ver8_1.pdf

20 years SPLE journey with SPL Community

Embedded Software Crisis (2000-)

Developing numerous software for product variations

- Applying microprocessor(and software) to electronic systems
- Developer can derive product variations only with change software

Figure 1: An overview of ECS.

Customer				А		В			
Engine Type		Di Inje	rect ection	Mu Inj	lti Port ection	Dire Injec	ect tion	Multi Port Injection	
Engine Size		1.6	2.0	1.0	1.6	2.0	3.0	1.0	2.0
Market	Japan	Х	Х	Х	Х	Х		Х	
	U.S.		Х		Х	Х	х		х
	Europe	Х	Х			Х		Х	х

Figure 2: ECS variants - An example.

Kentaro Yoshimura, Dharmalingam Ganesan, and Dirk Muthig. 2006. Defining a strategy to introduce a software product line using existing embedded systems. In Proceedings of the 6th ACM & IEEE International conference on Embedded software (EMSOFT '06). Association for Computing Machinery, New York, NY, USA, 63–72. https://doi.org/10.1145/1176887.1176897

Object-Oriented Embedded Software (2001-2003)

Challenge: How to Define "Object" to improve reusability

図 10 サブフレームワークの組合せ例

Fig. 10 $\,$ An example pattern of sub framework.

Kentaro Yoshimura, T. Miyazaki, and T. Yokoyama. 2005. A Model-based Development Method for Object-Oriented Embedded Control Systems. IPSJ Journal, Vol.46, No. 6, Information Processing Society of Japan, 1436-1446, 2005.

HITACHI

Inspire the Next

20 years SPLE journey with SPL Community

SPL Migration Approach for Legacy Systems (2006)

Defining a strategy with Cost model and legacy code analysis

Kentaro Yoshimura, Dharmalingam Ganesan, and Dirk Muthig. 2006. Defining a strategy to introduce a software product line using existing embedded systems. In Proceedings of the 6th ACM & IEEE International conference on Embedded software (EMSOFT '06). Association for Computing Machinery, New York, NY, USA, 63–72. https://doi.org/10.1145/1176887.1176897

20 years SPLE journey with SPL Community

SPLC Reports for local community (2007)

Sharing Research Activity for Community (Domestic)

Report on the 11th International Software Product Line Conference (SPLC2007)

TOMOJI KISHI,^{†1} MAKOTO NONAKA,^{†2} TOSHIAKI AOKI,^{†1} KENTAROU YOSHIMURA^{†3} and NATSUKO NODA^{†4}

This paper reports the 11th International Software Product Line Conference (SPLC2007), held in September 2007 in Kyoto. There were variety of presentations from basic researches to applications and experiences, and the conference provided practitioners and researchers a good opportunity to discuss together.

T. Kishi, M. Nonaka, T. Aoki, K. Yoshimura and N. Noda, "Report on the 11th International Software Product Line Conference (SPLC2007)", IPSJ SIG Technical Report, 2007.

SPLC Reports for local community (2011, 13, 17)

25

Sharing Research Activity for Community (Domestic)

SPLC 2011 @ Munich, DE

SPLC 2013 @ Tokyo, JP

SPLC 2017 @ Sevilla, ES

N. Noda, Y. Ishida, T. Kishi, M. Nonaka and K. Yoshimura, "Report of SPLC2011", IPSJ SIG Technical Report, 2011.

T. Kishi, M. Nonaka, T. Aoki, K. Yoshimura and N. Noda, "A Report on the 17th Software Product Line Conference (SPLC2013)", IPSJ SIG Technical Report, 2013.

N. Noda, H. Ogawa, T. Kishi, K. Kobata, T. Nakanishi, Y. Nishiura and K. Hayashi, "Participation Report of 21st International Systems & Software Product Line Conference", IPSJ SIG Technical Report, 2017.

Publishing Magazine Articles (2009, 2010)

Sharing Success Cases of SPL Practice for Society

T. Kishi, N. Noda, M. Inoki, M. Nonaka, K. Yoshimura, T. Kikuno and Y. Ishida, "A New Wave of Software Reuse- Widening Software Product Line Development", in IPSJ Magazine, Vol. 50, No. 4, 2009.

J. D. McGregor, D. Muthig, K. Yoshimura and P. Jensen, "Guest Editors' Introduction: Successful Software Product Line Practices," in IEEE Software, vol. 27, no. 3, pp. 16-21, May-June 2010, doi: 10.1109/MS.2010.74.

© Hitachi, Ltd. 2023. All rights reserved. 26

Introducing SPLE into Industry (2009)

Translated SPLE Textbook to Japanese

(訳)林好一,吉村健太郎,今関剛 (Yoshikazu Hayashi, Kentaro Yoshimura Takeshi Imazeki),
 "ソフトウェアプロダクトラインエンジニアリング — ソフトウェア 製品系列開発の基礎と概念から技法まで",エスアイビー アクセス (2009)

20 years SPLE journey with SPL Community

Technical Guidelines

Accelerate SPL application by developing guidelines based on SPL case studies and research results within the Hitachi Group

は果物をコア資産化することで、コア資産を拡充していく。 ・最初の1年でコア資産の構築と開発プロセス整備を行い、次年度から製品開発

確化。既存ソフトおよびプロセスの分析を行 開発プロセス整備を並行して実施した。

アクティビティ 実施順決定の考え方 に適用する計画で進めた。 ・者実に進めていくため、SPLレベル2を達成後、レベル3にレベルアップするよう 段階的にSPL導入した。 SPLL装施マップを参考に、具体的な作業項目を抽出し、担当や作業成果物を明 技術マップは、適用開始時には作成されてなかったので、参照していない。モラ ル指向開発手法を基盤にしたSPL開発のプロセスを留定して、それをもとにア・ ティビティと実施順を決めた。

Developing SPL Tutorial for Engineers

Integrating Domain Expertise and SPL Approach

Figure 3. Basic SPL Tutorial Format

K. Yoshimura, J. Shimabukuro, T. Ohara, C. Okamoto, Y. Atarashi, S. Koizumi, S. Watanabe and K. Funakoshi, "Key Activities for Introducing Software Product Lines into Multiple Divisions: Experience at Hitachi," *2011 15th International Software Product Line Conference*, Munich, Germany, 2011, pp. 261-266, doi: 10.1109/SPLC.2011.14.

Increase in the number of cases of SPL application

HITACHI

Inspire the Next

Maintaining SPL

32

Detecting Co-change Pattern from History (2008)

Kentaro Yoshimura, Fumio Narisawa, Koji Hashimoto, and Tohru Kikuno. 2008. FAVE: factor analysis based approach for detecting product line variability from change history. In Proceedings of the 2008 international working conference on Mining software repositories (MSR '08). Association for Computing Machinery, New York, NY, USA, 11–18. https://doi.org/10.1145/1370750.1370753

Maintaining SPL

Visualizing Code Clone Outbreak (2012)

K. Yoshimura and R. Mibe, "Visualizing code clone outbreak: An industrial case study," 2012 6th International Workshop on Software Clones (IWSC), Zurich, Switzerland, 2012, pp. 96-97, doi: 10.1109/IWSC.2012.6227883.

Maintaining SPL

Identifying Feature Constraints from Selection History (2010)

Fig. 1. Overview of our method

 Table 1. Example of feature selections

Product ID	feature 1	feature 2	feature 3	feature 4	feature 5
1	\checkmark			\checkmark	
2		\checkmark	✓		
3				\checkmark	\checkmark
4		\checkmark	\checkmark		
5		\checkmark	\checkmark	\checkmark	
6		✓		√	\checkmark
7				\checkmark	\checkmark
8	\checkmark			\checkmark	

 Table 3. Evaluation results

Classifications	# rules	%
Already identified in spec.	3	10%
Spurious relationships	7	23%
New feature constraints	21	68%
Total	31	100%

Application :

⇒ Recommendation system for selecting features "The product selected feature A&B also selected feature C"

Kentaro Yoshimura, Yoshitaka Atarashi, and Takeshi Fukuda. 2010. A method to identify feature constraints based on feature selections mining. In Proceedings of the 14th international conference on Software product lines: going beyond (SPLC'10). Springer-Verlag, Berlin, Heidelberg, 425–429.

INDUSTRIES AND SPL DEVELOPED PRODUCT VARIATIONS AND BE HAPPY EVER AFTER...

Here comes a new challenger!

New Challenger: Autonomous Systems (Robots)

- Autonomous robots are robots which can perform desired tasks in <u>unstructured environments</u> without continuous human guidance.
- An example: A Logistics robot recognizes item variations (packaging, positions) and picks and places item without human teaching.

Industrial Challenge : Autonomous Systems

Managing Variability of Environment : e.g. Robotics, Vehicles

- Modeling variability of operational design domain (ODD)
- Adapting/configuring systems for recognized environment (Dynamic SPL)
- Generating new variants for new environment

3. The Next

A Glance at the History of Software Engineering

'9	0 '0	0 '1	.0 '1	5 '20	D	
Society & Market	Generalization of calculators	Downsizing and price reduction of calculators	Web-enabled services	Globalization &Personalization	User Needs Diversification	
Technology	IT for Business Tasks	Digitalization of Equipment	Cloud Computing	Machine Learning	Generative AI	
Characteristic	Large-scale development	arge-scale Multi-Product High evelopment by Software u		Data-based decisions	General-purpose natural language dialogue	
	Planed control	Planed Reuse	Automation	Optimization	Collaboration	
Devel. Tech.	Object-Oriented, Dev. Process	ed, MBSE, CI/CD, DevOps, Dat SPLE Agile dev		Data-driven development	AI supported development	
		Deductive way		Inducti	ive way	
				© Hitachi, Ltd.	2023. All rights reserved. 40	

• We gather in-house specialists and actively promote utilization while managing the risks of "generative AI"

Is ChatGPT an expert of SPLE ?

HITACHI Inspire the Next

You are an expert of software product line engineering.

\$
. ×

 $^{\circ}$

Here's a brief overview of Key concepts and practices in SPLE: Domain Engineering (also called Feature Modeling) 1.Domain Engineering (d analyzing the 1. common and variable fe cs of a system. 2. **Application Engineering** They are two es. 3. Core Assets are modeled. 2.Apph Variability Management 4. nents from the application 5. Architecture domain. 6. Scoping 3.Core Assets: These ion across 7. Process Management multiple products in the 8. **Tool Support** 4.Variability Managem ts in the product line. Technique 9. **Benefits of SPLE** 5.Architecture: A produ to all Reuse 1 products. It's typically n reused, 2. Consistency configured, or replaced 3. Scalability 6.Scoping: Determining lves market 4. Efficiency analysis, technical feasi 10. Challenges 7.Process Management ation of SPLE. This often involves tech Upfront Investment 1. 8.Tool Support: SPLE be. 2. Complexity Int, and version control, among others. Ex 3. **Organizational Change** 42 Ltd. 2023. All rights reserved.

Could you describe how the generative AI can support the SPLE practioners?

Certainly. Software Product Line Engineering (SPLE) focuses on the systematic use of reusable assets to produce a family of related software products. Generative AI can play a significant role in aiding SPLE practitioners in multiple ways: Generative AI can play a significant role in aiding

SPLE practitioners in multiple ways:

Asset Reuse & Recommendations

Design & Architecture Suggestions

3. Automated Testing & Quality Assurance

Documentation & User Manual Generation

1.Feature Variation:

1. Generative Model

2.Asset Reuse & Recomm

- 1. Recommendation services) from a re development. This
- 3.Automated Testing & Qua
 - 1. Test Case Generat product variants, e
 - 2. Anomaly Detectio ensuring that new

4.Documentation & User Ma

2.

4.

5.

1. Given the feature com relevant documentation, user manuals, or even training material tailored to that specific configuration. © Hitachi, Ltd. 2023. All rights reserved.

its, his te this

res, or

erent

7. Predictive Analysis 8. Interaction Management

Domain Analysis and Scoping

9. Optimization

1. Feature Variation

10. Personalized User Experience

to produce

44

Try making variants with ChatGPT (Version 1)

Code Generation (CUI Chess) lass ChessPiece: def __init__(self, color, symbol): self.color = color self.symbol = symbol def valid_move(self, start, end, board): return True class Pawn(ChessPiece): def valid_move(self, start, end, board): dx = end[0] - start[0]dy = end[1] - start[1] if self.color == "white" and dx == 1 and dy == 0 and not board[end[0]][end[1]]: return True elif self.color == "black" and dx == -1 and dy == 0 and not board[end[0]][end[1] RNBQKBNR РРРРРРР • • rnbqkbnr Enter your move (e.g., 'a2 a3') or 'quit' to exit: return True return False

class Knight(ChessPiece):

alid move(self

Feature Description

This source code represents a basic implementation of a chess game, allowing pieces to be moved on an 8×8 board based on the standard rules of chess. Here's a breakdown of the features present in the code:

1. ChessPiece Base Class:

- A general representation of a chess piece with attributes for color (`white` or `black`) and symbol (for display purposes).
- Contains a general method `valid_move` that always returns `True`, but will be overridden by the specific pieces to implement the actual movement logic.
- 2. Pawn Class:
 - Inherits from 'ChessPiece'.
 - Implements the movement logic for the pawn piece in the `valid_move` method.
- 3. Rook Class:
 - Inherits from `ChessPiece`.
 - Implements the movement logic for the rook, which moves in straight lines either horizontally or vertically.
- 4. Knight Class:
 - Inherits from 'ChessPiece'
 - Implements the movement logic for the knight, which has an L-shaped movement pattern.
- 5. Bishop Class:
 - Inherits from 'ChessPiece'
 - · Implements the movement logic for the bishop, which moves diagonally.
- 6. Queen Class:
 - Inherits from `ChessPiece`.
 - . Implements the movement logic for the queen, which can move either like a rook or a

Try making variants with ChatGPT (Version 2)

Feature Description

In the provided Version 2 of the chess program, several notable updates and changes were made compared to Version 1. Here is a detailed description of the features and changes in Version 2:

1. GUI Integration using Tkinter:

- The `tkinter` library has been imported, and a GUI representation of the chessboard and its pieces has been added. This means that instead of interacting with the game through console inputs, the user will do so using a graphical interface.
- 2. New Class: `ChessGUI`:
 - The `ChessGUI` class represents the graphical interface of the chessboard.
 - Attributes:
 - 'board': Represents the chessboard object.
 - `root`: The main tkinter window.
 - `tiles`: A 2D list containing the buttons (tiles) of the board.
 - * `selected_piece_coords`: Stores the coordinates of a selected piece.
 - Methods:
 - `draw_board()`: Draws the chessboard by creating a grid of buttons (tiles). The color
 of each tile is set based on its position (alternating black and white).
 - 'update_pieces()': Refreshes the board display to update the position of pieces after moves.
 - `on_tile_click(x, y)`: Defines what happens when a tile on the board is clicked. If a piece is selected, it will attempt to move it to the clicked tile. Otherwise, it will select the piece on the clicked tile.
 - `run() `: Starts the tkinter main event loop to display the GUI.
- 3 Changes to the 'Board' Class

Consolidate two versions into organized components based on SPLE principles

\$

Software Product Line Engineering (SPLE) is a software engineering paradigm that aims at creating a collection of related products from a shared set of software assets instead of building each variant of the product individually. The main principles of SPLE include:

- Commonality and Variability Analysis: This is the process of identifying what parts of the software are common across all products in the product line (commonality) and what parts can change from one product to another (variability).
- Feature-based Development: Features represent units of functionality that are relevant to stakeholders and can be composed in various combinations to create products in the product line.
- Separation of Concerns: Different concerns of the software (like GUI, logic, data management) should be separated into different modules or components to make the system more maintainable and flexible.

Consolidate two versions into organized components based on SPLE principles

\$

Components:

- 1. Game Logic (Common Component):
 - Chess Pieces and their movement logic.
 - Chessboard representation and initialization.
 - Movement handling.
- 2. User Interaction (Variable Component):
 - Console-based interaction (from Version 1).
 - GUI-based interaction (from Version 2).

No, but

- A generative AI would be able to
 - analyze variability
 - draw the feature diagrams.
 - Implement variability.
 - generate test code
 - extract features from software.
 - more ...

Each task would be assisted or replaced by generative AI.

Need a new SPL process that is predicated on AI, not a partial replacement of the current process.

What if generative AI could build product lines **on demand** (when you need it)**?**

Traditional way:

Systematic

Preparatory

Deductive

- No need to proactive analysis, modeling, asset management, and ...?
- The roles between human and machine and the sequence of tasks will change.

1. Overview

Hitachi has promoted the application of SPLE through organizational activities, and have also contributed to the academic SPLE community.

2. Technical Journey

We have been promoting SPLE research mainly for embedded software development. Autonomous system is a new challenge for SPLE.

3. The next

Inductive technology; machine learning and generative AI may change SPLE. Especially, "on demand" capability of gen-AI would transform the SPLE process.

Hitachi Social Innovation is POWERING GOOD

We Are Hiring– Control & Automation Research Field

HITACHI Inspire the Next

Search [Hitachi R&D Recruiting] or contact us

20 years SPLE journey with SPL Community

