# 第 20 回 問題解決環境ワークショップ論文集

2017 年 8 月 29 日（火）ー　31 日（木）

# The 20th Problem Solving Environment

# International Workshop 2017

29-31 August 2017

問題解決環境(PSE) 研究会　2017

富士通サービシィーズ, 英国

Study Group on Problem Solving Environment 2017
Fujitsu Services, UK

22 Baker St, Marylebone, London W1U 3BW

Phone: +44-843-354-9654

# 第 20 回国際 PSE ワークショップ 2017 開催の挨拶

皆様のご支援とご協力のもと，記念すべき第 20 回 PSE ワークショップを今年は英国ロンドンにある富士通サービシィーズにて開催することができ，心より御礼申し上げます．現地にて欧州富士通研究所に勤務されている小橋博道氏に主催をお願いいたしました．富士通の皆様，小橋氏にお世話になりありがとうございます.

この第 20 回国際 PSE ワークショップ 2017 では，キーノート(1 件)、招待講演(1 件)と一般講演(5 件)を企画しております．キーノートといたしまして、英国グラスゴーにある University of Strathclyde から Sheng 教授をお招きし、”Challenges for numerical simulations in relativistic laser-plasma physics and applications: from attosecond pulse generation to laser fusion”と題しまして、ご講演いただきます。また招待講演といたしまして、欧州富士通研究所より、古屋リサーチフェローをお招きし、”Engineering Cloud”と題しまして富士通が展開しているシミュレーションクラウド基盤を紹介いただきます。

本ワークショップは既に 20 回目となり，その講演および研究内容の範囲は，植物工場関連の研究，シミュレーションの不確実性，情報科学教育支援，IoT 関連研究，拡張現実感など，第 1 回のワークショップから様々な分野へ広がりを見せています．講演や懇親会にて，参加者の皆様の活発な議論を楽しみにしております．また今後，以前は学生・院生であった現在ご活躍されている皆様方の卒業生の方々や，皆様の周りの企業の研究者の方々にご参加いただき，問題解決環境ワークショップの活動の維持発展にご支援，ご協力を切にお願い申し上げます.

最後に，本ワークショップ Web 運営を担当いただいた宇都宮大学川田研究室の加藤寛樹氏，論文誌の作成等，本ワークショップ開催の全面的なバックアップをして頂いた欧州富士通研究所の小橋博道氏に感謝します.

2017 年 8 月 29 日

主催　PSE 研究グループ

Study Group on Problem Solving Environment 2015 Organizing Committee Members

USAMI Hitohide, UMETANI Yukio, KASHIYAMA Kazuo, KADOOKA Yoshimasa, KAWATA Shigeo, KOJIMA Yoshitaka, TERAMOTO Takayuki, NIWA Kazuhisa, HAYASE Yoshikazu,　BARADA Daisuke, HIOKI Shinji, MAEDA Taiyo, MATSUMOTO Masami, MANABE Yasuhiko, MIYACHI Hideo, MURATA Tadahiko, HWANG Soonwook

# Program

| | |
|---|---|
| **Day 1: 2017-08-29** | |
| 13:50 | Opening<br><br>Dr. Hiro Kobashi |
| 14:00 | **Invited Special Talk**: "Engineering Cloud (tentative)"<br><br>Ms. Sachiko Furuya, Research Fellow, Fujitsu Laboratories of Europe |
| 15:10 | "Robots as educational PSE"<br><br>Prof. Shinji Hioki, Tezukayama Univ, Japan |
| 15:40 | "Development of PSE System to Enable Edge Computing for IoT<br><br>（IoT のためのエッジコンピューティングを可能とする PSE システムの開発）"<br><br>Masami Matsumoto, National Institute of Technology, Yonago College |
| 18:00 | Workshop Dinner |
| **Day 2: 2017-08-30** | |
| | Workshop Tour Bletchley Park<br><br>https://www.bletchleypark.org.uk/ |
| **Day 3: 2017-08-31** | |
| 09:00 | **Keynote Speech**: "Challenges for numerical simulations in relativistic laser-plasma physics and applications: from attosecond pulse generation to laser fusion" (tentative)<br><br>Prof. Dr. Zheng-Ming Sheng, University of Strathclyde, Glasgow, U. K. /Shanghai Jiao Tong University, Shanghai, China |
| 10:10 | "A Review of PSE (Problem Solving Environment) in Computing Science "<br><br>Prof. Shigeo Kawata, Utsunomiya University, Japan |

| | |
|---|---|
| 10:40 | "Data reduction by software point rendering"<br><br>Prof. Hideo Miyachi, Tokyo City University, Japan |
| 11:10 | “Dixiter: A card game AI of Dixit”<br><br>Dr. Hiro Kobashi, Fujitsu Laboratories of Europe, UK |
| 11:40 | Final Remark<br><br>Prof. Shigeo Kawata |

# Robots as educational PSE

Shinji Hioki
Department of Business Administration, Tezukayama University, Nara, JAPAN,
hioki@tezukayama-u.ac.jp

In order to maintain learner's motivation, we propose the use of "Robots" as efficient educational PSE tools. As a test, we adopt "UNIBO" robot produced by the Japanese company "unirobot.com". To tackle the questions spoken by the robot, the learner might keep his motivation. We made a simple application to create source code for the robot.

***Key Words:*** Robot, AI, Artificial Intelligence, e-learning, PSE

## 1. Introduction

In order to achieve good performance of self-learning, a lot of mechanisms have been proposed and tested. "e-learning" can be the most efficient tool. It can be performed without the restriction of time and place.

However, "e-learning" alone cannot produce effective results. Because some learners cannot keep their motivation without any human advice and support.

The blended learning which is the mixed version of the normal classroom (with human advice and support) and e-learning (self-learning) has revealed to be effective compared with the e-learning only.

Fig.1 shows the correlation between the scores of the e-learning (pre-/post study in the figure ) and the final score.
( in my classroom in Tezukayama University)

So if learners can keep their motivation in e-learning environment (and/or in self-learning environment) without any human resource, good educational effect will be expected.

In this paper, we will make a trial with the use of "Robots" as educational PSE tools.

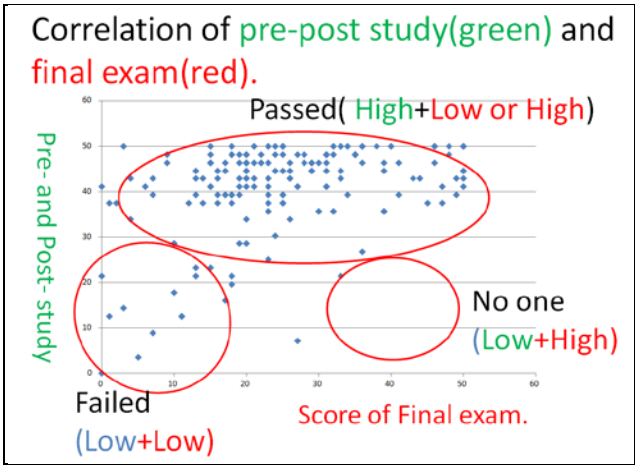


Fig.1 Correlation between e-learning and final score.

## 2. Robots

Recently many robots become available. We must decide on of them, because we cannot buy many of them.

Around the reasonable price, we first select the following 4 robots as candidates. Then we ask many students in our university which robot looks friendly.

(1) OHaNAS

OHaHAS[1] is a robot treated by “TOMY Company, Ltd.” (Fig.2)

Fig.2 OHaHAS

OHaNAS can be used with the Smart-Phone application. If we ask something to OHaNAS, he will reply and show results on the display of the Smart-Phone.

(2) PARO

PARO[2] is a robot treated by “ DAIWA HOUSE INDUSTRY CO.,LTD. (Fig.3)


Fig.3 PARO

PARO is the world's most therapeutic robot certified by Guinness World Records.

(3) Sota

Sota[3] is a robot treated by “Vstone Co.Ltd”.(Fig.4)

Fig.4 Sota

Sota is developed by Prof. Ishiguro in Osaka University and can be used on the desktop.

(4) Unibo

Unibo[4] is a robot treated by “Unirobot.com”.(Fig.5)

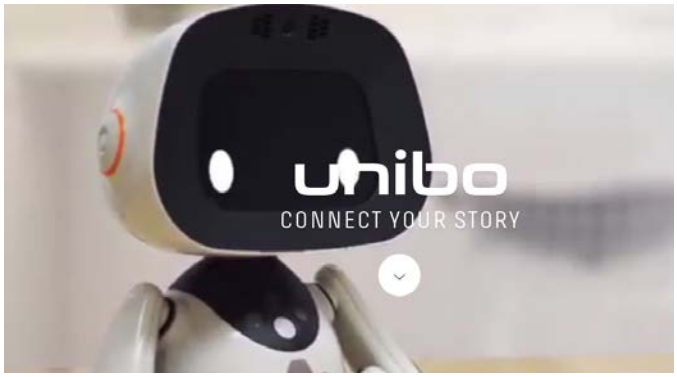


Fig.5 unibo

Unibo is an AI( Artificial Intelligence) robot which is thought to be social for us. Unibo can be programmable easily.

We show many students (faculty: economics, business and psychology ) the characteristic of these interesting robots and the differences among them. Then we ask students to select one of them according to their preference. We think this point is very important because the student can keep their motivation with the preferable robots. The result is in Table.1.

| robot | economics | business | psychology | total |
|---|---|---|---|---|
| OHaNAS | 3 | 13 | 30 | 46 |
| PARO | 12 | 10 | 33 | 55 |
| Sota | 4 | 6 | 9 | 19 |
| Unibo | 23 | 19 | 28 | 70 |
| # of sample | 42 | 48 | 100 | 190 |

Table.1

According to this result, we have decided to use “Unibo” as an educational PSE for the first time.

## 3. “Unibo” programming

In unibo programming, GUI environment is available as in Fig.6.

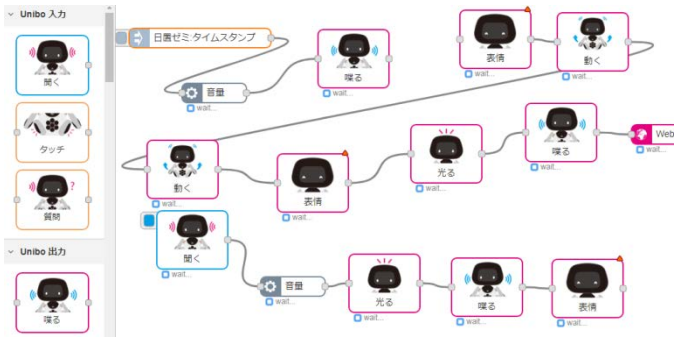

Fig.6 Unibo GUI programming

Recently many programming environment have been available for beginners. Almost of them adopt GUI (Graphical User Interface) as an interface.

Flow chart in Fig.7 can be realized in Unibo programming environment as Fig.8.

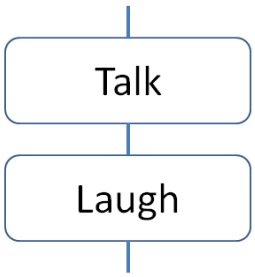


Fig.7 Flow Chart of Talk and Laugh

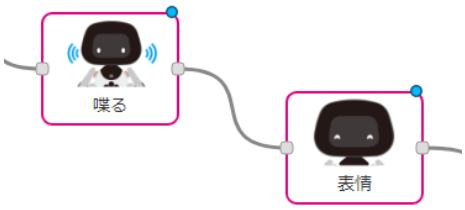


Fig.8 Unibo GUI Programming

Thus the program flow can be built in schematic way by the use of the connected line between two objects.

GUI is very simple and easy to use, but it costs a lot of time if we make same routines repeatedly.

In the Unibo native language, this connected line is realized as an unique label shared by two objects. So we have developed the application which generates Unibo program.

When we make “if / else” statement in Fig.9, in Unibo environment it looks as in Fig.10.

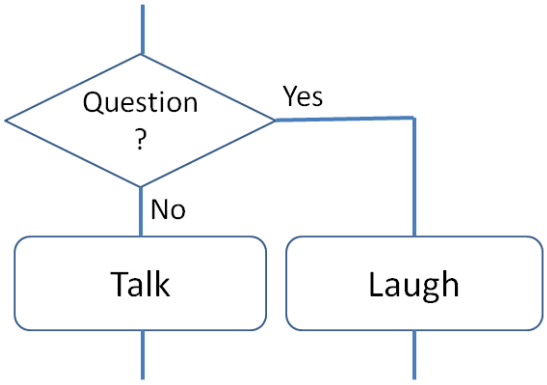


Fig.9 if-else statement in Flow Chart

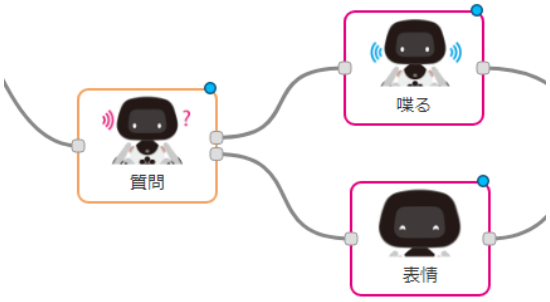


Fig.10 if-else in Unibo GUI

This if-else flow can be realized in Unibo native language as in Fig.11.

```
name: A , type: question , connected: B , C
name: B , type: speech   , connected: D
name: C , type: laugh    , connected: E
```

Fig.11 if-else in Unibo native language

We have made Web-based application which creates a Unibo native program.(Fig.12)

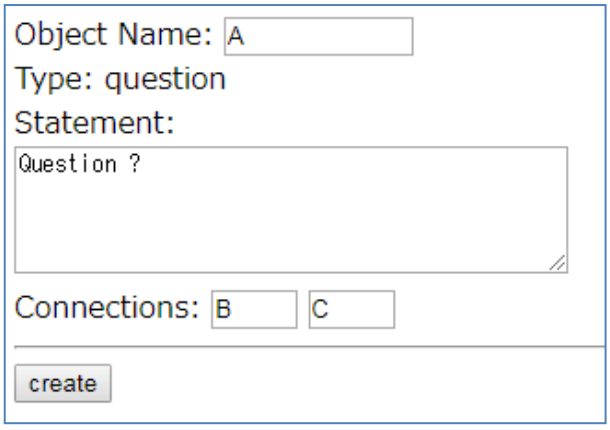


Fig.12 Web-based application


Fig.13 Unibo face

When we run this program, Unibo speak "Question ?" and display on his face. (Fig.13)

If we use this repeatedly, we can easily make a "perfect drill" by which one cannot finish it unless one answers all questions correctly.(Fig.14)

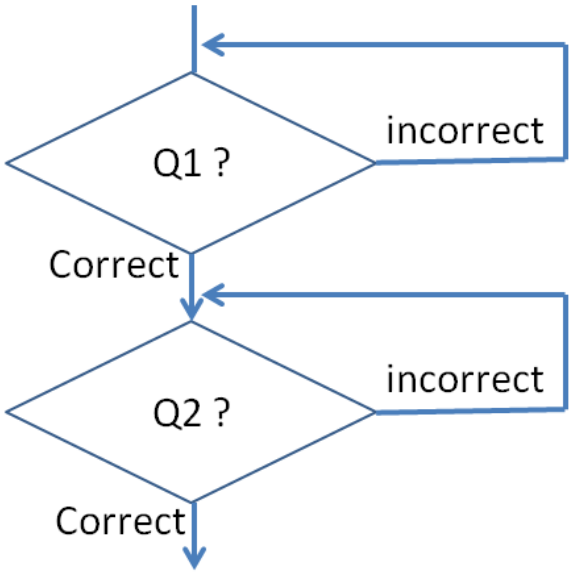


Fig.14 perfect drill

## 4. Summary and Outlook

We have prepared the tools in Unibo programming. Next step is to check the effectiveness of this scheme.

We hope that Unibo program is useful to maintain learner's motivation and the use of "Robots" becomes an efficient educational PSE tools.

## *REFERENCES*

[1] http://www.takaratomy.co.jp/products/omnibot/ohanas/

[2] http://www.daiwahouse.co.jp/robot/paro/

[3] https://sota.vstone.co.jp/home/

[4] https://www.unirobot.com/

# Development of PSE System to Enable Edge Computing for IoT

Masami Matsumoto
Department of Electrical and Computer Engineering, National Institute of Technology, Yonago College, 4448 Hikona-cyo, Yonago City, Tottori Prefecture, Japan 683-5042, E-mail: matsu@yonago-k.ac.jp

**Abstract:** This paper presents the PSE system for Internet of Things (IoT). The purpose of this study is to improve the accessibility of the objects network and system scalability. The IoT system collects and manages sensor data via the Internet, performs remote control and statistical analysis, and creates new value of the system. Therefore, it is important to constantly connect to the Internet and analyze and visualize various types of big data. However, building an IoT network requires extensive knowledge of ICT technology. The developed system supports edge computing which performs data processing in the middle layer of the network. This system reduces the load of data transfer, and it was possible to support the construction of a problem solving environment (PSE) using various information systems by IoT.

**Key Words:** IoT; PSE; Edge Computing

## 1. Introduction

In recent years, sensors with communication functions other than PC and "Internet of Things (IoT)" connecting various devices to the Internet have been practiced. IoT system collects and manages data measured with sensors via the Internet, and can generate new value by performing remote system control and statistical analysis. In order to construct the IoT system, it is necessary to have an environment that always connects things to the Internet, but the following is another problem 1-2).

- Knowledge of the IoT network configuration is diverse and difficult.
- An explosive increase in the amount of data and traffic concentrate on the cloud.
- Sensor specifications and data format are complicated.
- Standard data visible environment required.
- Secure communication is required.
- Communication power consumption is large.

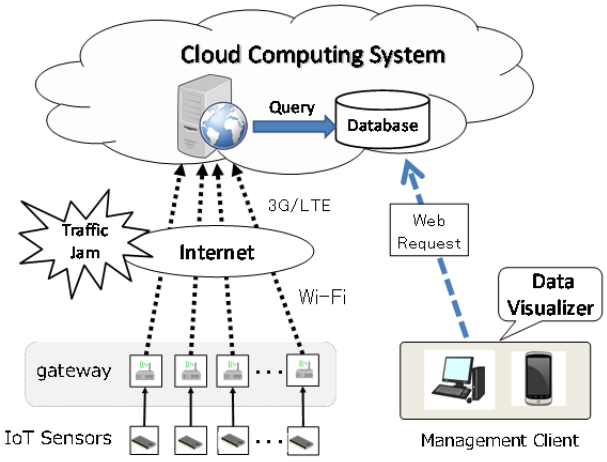


**Fig. 1.** Cloud Connection Technology for IoT

In order to solve above problems, there is a need for a Problem Solving Environment (PSE) that supports the construction of IoT, such as providing a network and a visualization environment. Figure 1 shows the IoT system connecting to the cloud. That is a problem that traffic concentrates on a standard system using a cloud. In this IoT system, the sensor is directly connected to the cloud via a dedicated gateway. This network structure has the above problems. With this network configuration, additional sensors will increase traffic to the cloud and increase data loss.

## 2. PSE System for Internet of Things (IoT)

This research proposes the IoT system configuration shown in Figure 2. In this IoT system, an edge which is an intermediate layer is provided between the cloud and the gateway and the sensor data is sent to the upper layer collectively. Furthermore, using IEEE 802.15.4 (ZigBee) with high reliability for communication between the gateway and the cloud, it is possible to prevent data loss due to communication failure. By adapting the edge database structure to various sensors, a more flexible system can be realized.

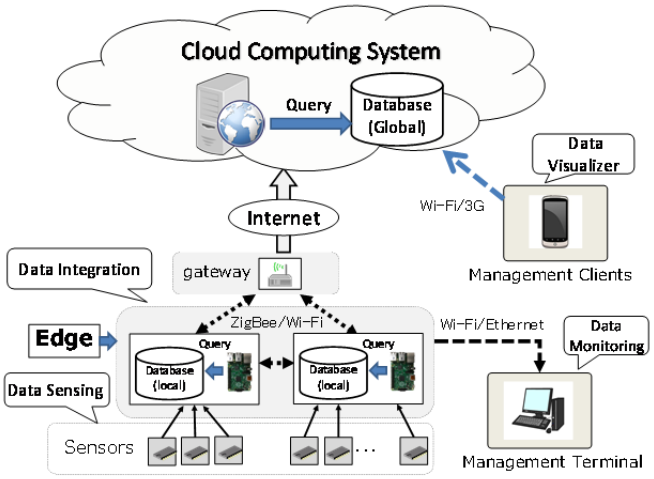


**Fig. 2.** Improvement of IoT system by edge computing

In addition, the edge uses low power consumption communication standards, and power saving of edges and sensors is performed.

## 3. Edge Computing System

This system consists of the sensor module, edge and Gateway. This framework provides an IoT system that can easily develop a network composed of an automatic configuration database system. By implementing the network function at the edge, it provides users with sensor data collection by fixed or moving and Web-based visualization environment. The configuration of the sensor and edge system is shown in Figure 3. At the edge, data is stored in an internal database.

Raspberry Pi is used at the edge of this system 3). The Raspberry Pi module is a credit card-seized single-board computers developed in the UK by the Raspberry Pi Foundation. The Raspberry Pi 3 Model B features a quad-core ARMv8) running at 1.2 GHz with 1 GB of RAM. Linux OS is installed in Raspberry Pi module. A Linux-based system is supported many functions such as Virtual Memory System (VMM) and the multi-task operations. Raspbian is a free operating system based on Debian optimized for the Raspberry Pi hardware 4). It is controlled a Wi-Fi network thorough a TCP/IP socket. The ZigBee wireless communication module TWE-Lite is also used 5).

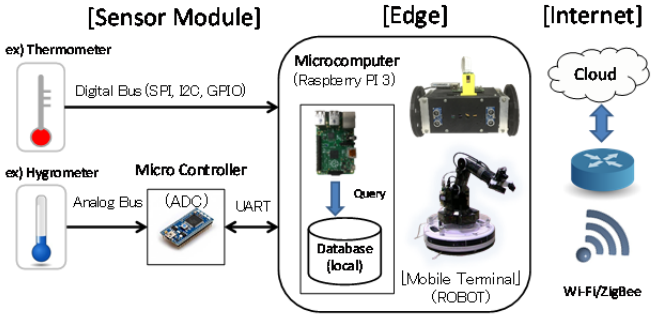


**Fig. 3.** Connection between Sensor and Edge System

### 3.1 Sensor Module

The feature of this system is that the edge for collecting data is placed between the sensor device and the cloud system. The sensor device measures data in various formats such as analog data such as temperature and humidity, vital data, and camera image. The measured data is converted from an analog value to a digital value and then transmitted to the edge.

Recently, it became possible to connect sensor hardware with GUI based on Web-based system such as Node-Red 6). This program is a programming tool for wiring together hardware devices, APIs and online services in new and interesting ways.

### 3.2 Data Collection on the Edge

The data measured by the sensor module is transmitted to the edge after conversion from the analog value to the digital one. The edge saves the received data in an internal database. Data Transmission on the Edge.

For data merged at the edge, ZigBee/Wi-Fi communication is performed and multiple edge data are merged again. ZigBee is a low-power protocol compared to Wi-Fi and 3G. However, the signal level is low and the communication distance is short. By incorporating a communication module in a movable item, data integration function is implemented.

The edge saves the data received from the sensor module in the database. The configuration of the edge system is shown in Figure 4. Programs of this system are written using Go language 7).

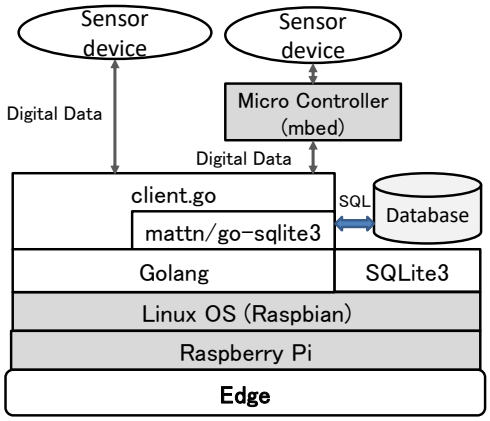


**Fig. 4.** System Configuration of the Edge

### 3.3 Database System on the Edge

In order to deal with sensors of various standards, it is necessary to optimize the table structure according to the type of data. A data table for storing the data received from the meta-table that associates measurement data table produced on the edge. Figure 5 shows the table structure of the device database.

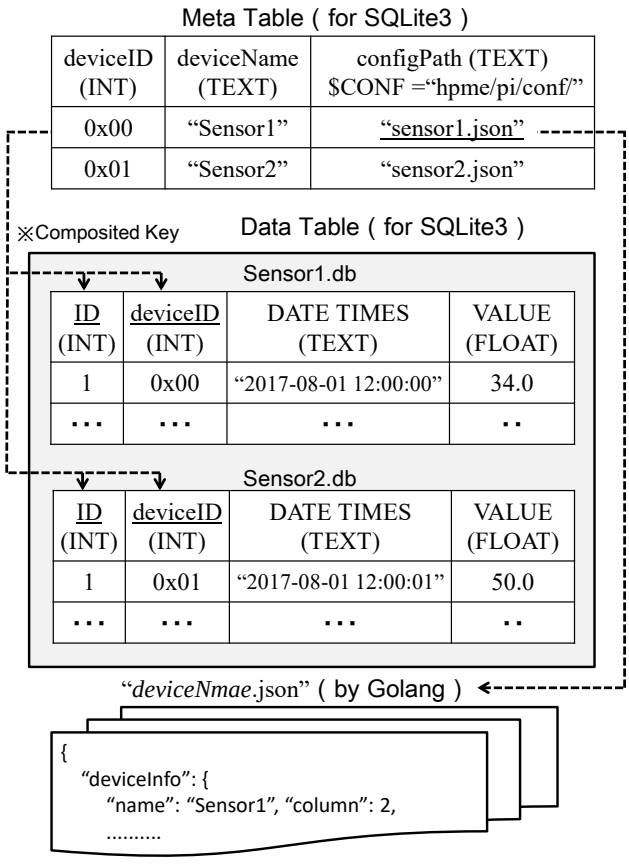


**Fig. 5.** Configuration Files of the Device Database Tables

The structure of the data table is defined by the configuration file. The Meta Table has a target of linking the data and the structure of the data table, and it is composed of a plurality of fields. The configuration file defines the data type, etc. of the table names and column number and each column. The configuration file on Figure 5 is an example of a setting file of the sensor device tables. Table names and column number, the type of each column is defined in the configuration "JSON" file. The data table is automatically generated this setting file by using Go language program at the edge module. Each device is assigned an unique ID to identify the table.

The system was lighter and faster using the SQLite database management systems Version 3 (SQLite3) 8). The SQLite3 database is a single ordinary disk file that can be located anywhere in the directory hierarchy. It is compatible with the IoT system because its data type is flexible and the data size can be easily expanded. In addition, a public domain is not required license of the core source code. SQLite 3 does not require password setting and installation cost is low. However, it is necessary to strengthen security.

## 4. Secure Data Collection Framework

IoT systems consist of connected "things" and networks and thus should be regarded as an integrated system of ICT with physical components. It is important to ensure physical safety in addition to existing information security measures. It is essential that the IoT system be designed, developed and operated in principle as emphasizing security. This PSE framework aims to clarify basic and essential security requirements of a secure IT system.

Hacker attacks against Web server applications are targeted at database management systems (DBMS). In the previous chapter, the implementation of the DBMS without the administrative authority setting function is shown. Therefore, as a security measure, a mechanism used for crypto currency was applied to access and operation of IoT data 9).

### 4.1 Go Ethereum

Block chain technology used in crypto currency is based from Decentralized Autonomous Organization (DAO) algorithm.

The system of Figure 6 is equipped with an IoT data collection function, and realizes a secure Ethereum protocol. 10). Go Ethernet is the original implementation of the Ethernet protocol. It is pure Go application, fully open source and Licensed under the GNU LGPL v3. The latest stable version at the time of writing this paper is 1.6.7. The constructed system operates with Version 1.4.x or higher.

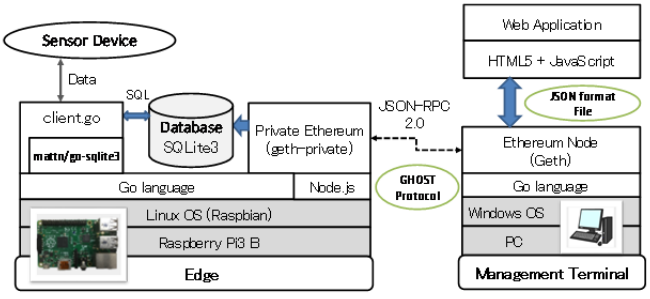


**Fig. 6.** Go Ethereum System

Go Ethereum is a decentralized platform that runs smart contracts, applications that run exactly as programmed without possibility of downtime, censorship, fraud or third party interference 11). A Smart Contract is that a predetermined process is automatically executed when a certain condition is satisfied.

### 4.2 Private Network Construction Using Geth

Go Ethereum, to build a private network, and displays the data converted into JSON format to operate the client Geth on the management terminal. Go Ethereum command line interface (CLI) "geth" is an Ethereum command. Go Ethereum is available ether as a standalone client called "Geth". Where "Ether (ETH)" is a Unit of virtual currency.

By installing "Geth", it is possible to perform the following operations.

- Mining and Remitting of ether
- Generate Smart Contract
- Generate transaction
- Confirm block chains

On this IoT system, a private network is constructed using Ethernet to acquire ether. The command Geth has an interactive console.

At the first, define the initial block information of the block chain in the "genesis.json" file. Then, the first block of the chain into "Genesis Block". Figure 7 shows the console screen lists at the time of initialization. A hash value based on the account is calculated and displayed as an address. The Geth starts up as a console application. It connects to the default port 8545 network.

The edge is attached to the network without using the console. To collect data by accessing the other edge is connected from an application using JSON-RPC 2.0 protocol. In this system, interface programs based on CLI created in Go Language are used to support various platforms. To access the block chain must authenticate with the address and passphrase generated at each edge, the intrusion protection.

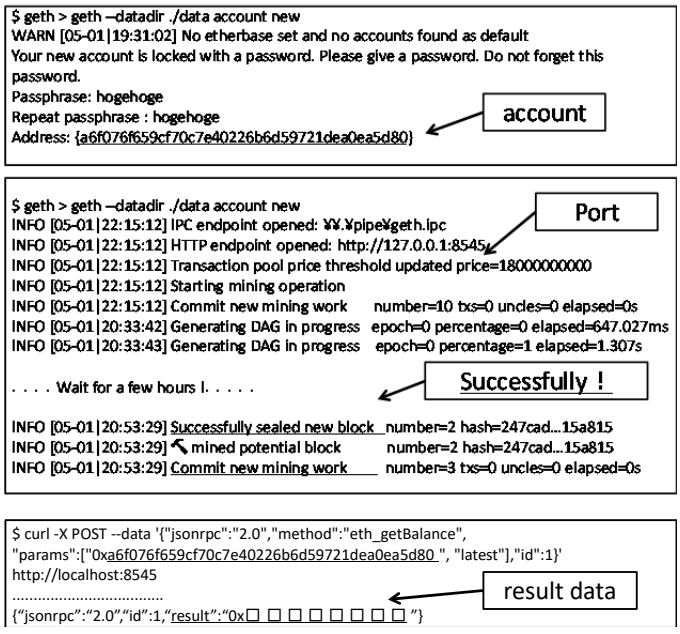


**Fig. 7.** Console Message Lists of the new Genesis Block

## 5. Data Integration Performance

The edge of the system moving goes to collect data. It is important to reduce the time required for the edge to collect data. This section evaluates the performance of the data collection system. As a result of measuring 1,000 actual data (36 bytes / case), the average collection time per data was 64.7 ms, and the standard deviation was 2.78 ms. Therefore, it is confirmed that this system has performance capable of collecting and integrating 15 edges per second if it is less than 36 bytes 12). Figure 8 shows the data transfer rate versus respect to the number of edges.

The purpose of the research is to develop a PSE system aimed at supporting the construction of the IoT system.

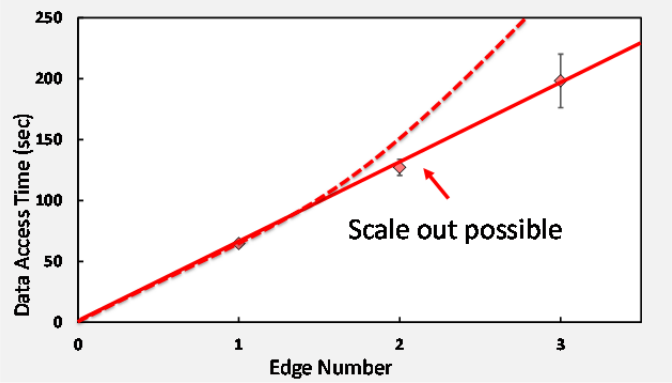


**Fig. 8.** Data Access Speed vs. Edge Numbers

## 6. Data Visualization

By providing a function as a Web server to the edge, it performs visualization of control and data nodes by the management terminal. The management terminal accesses the edge through the Web browser. The management terminal can monitor data using the edge movement control function and graph display function of the collected data.

Furthermore, implementing the Web-based data visualization service function on the edge. This function enables graphing and high-speed monitoring from the Web browser of the management terminal on the LAN. Figure 9 shows an implementation of the data graphs by the sensor data viewer. Performed automatically scaled in conjunction with the sensor list database and graphed using JavaScript. This function reduced communication cost and realized seamless data visualization between cloud environment and local system.

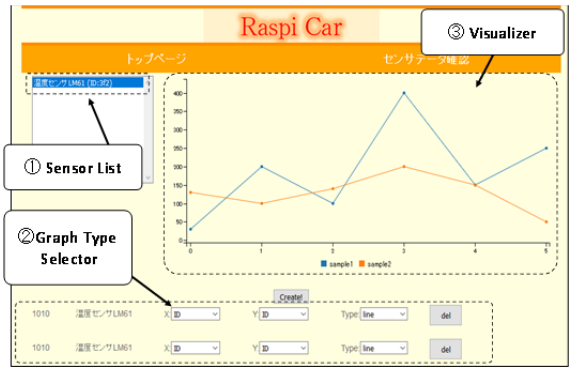


**Fig. 9.**Visualization environment User interface (JavaScript)

## 7. Conclusion

In this study, IoT PSE system is developed to support IoT construction. This developed PSE system has a network function for movable edges, and implements automatic sensor data collection function and Web-based visualization environment.This system semi-automatically constructs a system with high flexibility by making the structure of a database storing sensor data correspond to a wide variety of sensors. It became possible to reduce the amount of traffic with the cloud by integrating the measured sensor data at the edge after accumulating it at the edge. It does not require a repeater to the network for collecting sensor data on the edge to be moved. This reduced the cost of the IoT network.

By using this PSE system, support for constructing problem solving environment using various information systems by IoT became possible.

## *Acknowledgments*

I wish to thank the timely help given by Daisuke Yamane of Cybertrust Japan Co., Ltd. in analyzing the large number of samples.

## *REFERENCES*

1) M. Matsumoto, “Development of the Environmental Management Support System Using the Sensor Network”, Proceedings of the 19th JSCES, Vol. 19, F-11-6.
2) M. Matsumoto, “Development of AR Visualization System of Sensor Data”, Proceedings of the 21st JSCES, Vol. 21, E-11-5.
3) Raspberry Pi Foundation, http://www.raspberrypi.org/
4) Raspbian, https://www.raspbian.org/
5) TWE-Lite, https://mono-wireless.com/
6) Node-Red, https://nodered.org/
7) Go languege, http://golang-jp.org/
8) SQLite, https://sqlite.org/
9) The Bitcoin Foundation, https://bitcoinfoundation.org/
10) go-ethereum, https://ethereum.github.io/go-ethereum/
11) Ethereum Foundation, https://www.ethereum.org/
12) M. Matsumoto, D. Yamane, “Development of PSE System for IoT Framework”, Proceedings of the 22nd JSCES, Vol. 22, D-12-6.

# A Review of PSE (Problem Solving Environment) in Computing Science

Shigeo Kawata[1] and Yukio Umetani[2]

[1] Graduate school of Eng., Utsunomiya University (Yohtoh 7-1-2, Utsunomiya 321-8585, Japan)
[2] Emeritus Professor, Shizuoka University (Ohya 836, Surugaku, Shizuoka 422-8529, Japan)

In this paper a review is presented on the PSE (Problem Solving Environment) concept in computational engineering and science. In the PSE concept, a part of application of solutions, which can be solved mechanically, is performed by computers or machines or software. PSE provides integrated human friendly innovative computational services and facilities for easy incorporation of novel solution methods to solve a target class of problems. PSE is an innovative concept to enrich our e-Science, e-Life, e-Engineering, e-Production, e-Commerce, e-Home, etc. The PSE-relating studies were started in 1970's to provide a higher-level programming language than Fortran, etc. in scientific computations. The trend at that time was resulting in PSE, CAE (Computer Assisted Engineering), libraries, etc. At present PSE covers a rather wide area, for example, program generation support PSEs, education support PSEs, CAE software learning support PSEs, Grid/Cloud computing support PSEs, job execution support PSEs, e-Learning support PSEs, etc. This review paper includes the PSE definition, a brief history of PSE, example PSE study activities, uncertainty management PSE and a future research directions in PSE.

***Key Words:*** Problem Solving Environment, Computer assisted computations, Scientific simulation, e-Science, PSE, Uncertainty in scientific computing

## 1. Introduction

Problem Solving Environment (PSE) concept provides integrated human-friendly innovative computational services and facilities for easy incorporation of novel solution methods to solve a target class of problems. For example, a PSE generates a computer program automatically to solve differential equations[(1)~(12)]. Now the PSE concept covers rather wide areas in our society. PSE is an innovative concept to enrich science, human life, engineering, production and our society toward a programming-free environment in computing science. In the PSE concept, human concentrates on his/her target problems and works on solutions, and a part of application of solution, which can be solved mechanically, is performed by computers or machines or software. At present many kinds of computer-assisted problem solving environments are found everywhere in our life and in our society.

On the other hand, even today human power still contributes greatly to develop and write new softwares. For example, in scientific researches scientific discoveries are supported by theory, experiments and computer simulations. New researches tend to require new computer programs to simulate phenomena concerned. In another example, in developing new products engineers would also need new computer programs to develop the products cost-effectively. They may have to develop the new programs or learn how to use the programs for the product development. New services may also need new software systems. Therefore, the researchers and engineers may write or develop the new computer programs or learn how to use the programs. They do not like to develop nor learn the computer programs to solve their problems, but they like to devote their efforts to solve their target problems themselves.

In addition, computer simulations became the third important method after theoretical and experimental methods in science and engineering. Computer assisted problem solving is one of key methods to promote innovations in science and engineering, and contributes to enrich our society and our life from scientific and engineering sides.

The PSEs have provided the new directions to support users, engineers and scientists for developing new services and new softwares, and also for solving their target problems based on computer systems.

The PSE-relating studies were started in 1970's to provide a higher-level programming language than Fortran, COBOL, ALGOL, PL/I and others in scientific computations. The trend at the time was reasonable to deliver more human-friendly programming environment beyond the higher-level languages shown above. Then the PSE research activity was started as well as activities of Computer Assisted Engineering (CAE) and software libraries. After the intensive developments in computer hardware and also in computer algorithms, researchers and engineers had expected an innovation in program writing and developing power. However, the enhancement in the programming power was relatively slow and weak compared with the enormous evolutions in the present hardware and algorithm power enhancements.

At present PSE covers a rather wide area, for example, program generation support PSE, education support PSE, CAE software learning support PSE, grid/cloud computing support PSE, job execution support PSE, learning support PSE, uncertainty management in scientific computing, PSE for PSE generation support (PSE for PSE), etc.

The paper includes a brief history of PSE, example PSE study activities and a future of PSE.

## 2. Computer-Assisted Problem Solving Environment (PSE)

PSE is defined as follows: "A system that provides all the computational facilities necessary to solve a target class of problems. It uses the language of the target class and users need not have specialized knowledge of the underlying hardware or software"[(6)]. PSE provides integrated human-friendly innovative computational services and facilities to enrich science, life, engineering, production, commerce and our society. Based on the PSEs, human concentrates on target problems themselves, and a part of solution is performed mechanically by computers or machines or software.

In computing sciences, we need the computer power, the excellent algorithms and the programming power in order to solve scientific problems leading to scientific discoveries and

development of innovative new products and services. So far, the computer power and the computing algorithms have been developed incredibly, and have provided enormous contributions to sciences, productions and services. On the other hand, the programming power has not been developed well. The concept of PSE was proposed to support the programming power in science and engineering, and has been explored for decades.

In 1985, IFIP (International Federation for Information Processing) WG2.5 (Numerical Software)[16] organized a working conference on PSE and published the proceedings[17]. In 1991, a working conference on Programming Environments for High-Level Scientific Problem Solving was held[18]. In addition, a book on PSE was also published[19]. In 2007, a working conference on Grid-Based Problem Solving Environments was held[20]. A working conference on Uncertainty Quantification in Scientific Computing was held in 2012[21]. The PSE activity including scientific computing environments is one of research projects in IFIP WG2.5[16]. In these decades, international conferences tend to include the topic of PSE as one of standard topics in scientific computing. It has been recognized that PSE is an important research area in scientific computation and high-performance computing. In parallel, the PSE activities have started in several societies, scientific groups and countries. For example, in Japan, the PSE research group started in 1998 based on the previous individual PSE study activities, and the Japan Society for Computational Engineering and Science (JSCES) started in 1995, including the Study Group on PSE[22], [23].

The PSE studies have been extensively explored over the past few decades. The explorations have been supported by the reinforced computer power and algorithm power. PSE has boosted the programming power, and have enriched problem solving methods in science and engineering to bring us innovations.

One of PSE studies[1]~[12] has been extensively explored in order to support engineers and scientists to compute or solve their own problems based on for partial differential equations (PDEs), for example. One of the major objectives in PSE researches is to help users compute or solve their problems without heavy tasks, for example, without complete knowledge for computations[24] and/or the programs used. In this sense, the PSE provides an infrastructure for software for computational engineering and sciences.

One of typical PSEs for PDEs-based problems is ELLPACK[8], [24]. ELLPACK is a high level system for solving elliptic boundary value problems. One can solve routine problems by simply writing them down and naming the methods to be used. The ELLPACK high-level language can reduce the programming effort for a "routine" elliptic problem.

Another typical PSE for PDEs-based problems is DEQSOL[7], [10], [11]. DEQSOL was designed to develop an easy-to-use system for problem solving of PDE-based problems by finite difference method and finite element method. DEQSOL creates optimal Fortran codes oriented to the Hitachi vector processors.

Another PSE system of NCAS[1], [2], [4], [9] inputs a problem information including PDEs, initial and boundary conditions, and discretization and computation schemes, and outputs a program flow graph, a C-language source code for the problem and also a document for the program and for the problem (see Fig. 1). On a host computer a user inputs his/her problem, and NCAS guides the user to solve the problem. The distributed PSE for PDEs consists of several modules: problem description, discretization, equation manipulation, program design, program generation, documentation support, module liaison and job execution service. Each module is distributed on distributed computers. Each distributed module communicates with the host module, so that outputs from each module are visualized. Independent modules, which are developed by other engineers or users for one of the functions specified above can also be used after adjustments to the distributed PSE interface[25], if necessary. The module liaison module generates an adapter module for the distributed PSE modules. The adapter module generated by the module liaison system inputs output data from preceding modules and/or external modules, and connects the data to the input data for the next module. The PSE contains all the information of the problem, PDEs, discretization scheme, mesh information, equation manipulation results, program design structure, variables and constant definitions and program itself. Therefore, the documentation support module also generates a document for the program generated together with the problem itself in the PSE[26].

A PSE module in NCAS also helps users generate MPI-based parallel simulation programs based on PDEs[4]. The NCAS capability explores possibilities to visualize and steer the parallel program design process. At present NCAS supports a domain decomposition in a design of a parallel numerical simulation program, and the domain decomposition is designed or steered by users through a visualization window. After designing the domain decomposition, the parallel program itself is also designed and generated in NCAS, and the designed parallel program is visualized and steered by a Problem Analysis Diagram (PAD). In NCAS, MPI functions[27] are employed for message passing, and a single program multiple data (SPMD) model is supported. The visualization and steering capabilities provide users a flexible design possibility of parallel programming.

Some PSEs provide a job execution support service on cloud or grid[28], [29]. It is difficult for users to submit jobs to distributed computers and to retrieve calculation data from them in scientific computing. A robust job execution service system was also developed in a closed distributed computer system[28]. The job execution service system consists of dynamic system management servers, execution servers and data servers. The

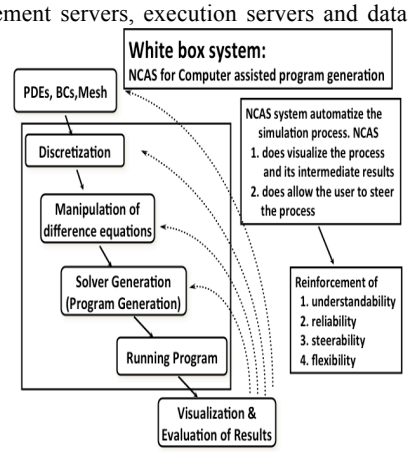


Figure 1. An example PSE for computer assisted scientific program generation support: NCAS. NCAS inputs partial differential equations (PDEs), initial and boundary conditions, discretization method and algorithm, and outputs a C language program. The PDEs are automatically discretized and the program is generated mechanically. NCAS is a white box system, in which users can see and steer all the processes of program generation. NCAS system contains all the information for program generation, including basic equations, discretization schemes, discretized equations, boundary and initial conditions, mesh structure, program structure, and definitions of variables and constants. Therefore, a document for the corresponding program is also generated together with the program itself.

dynamic system management server is duplicated in order to keep the system robust, and has an assistant management server. The dynamic system management server has a function of the job execution system management, including software deployment, program compilation, job execution, job status retrieval and computing data retrieval. This system does not require special middleware such as Globus[30] or UNICORE[31] or so. Users access the web page on the dynamic system management server, and the clients submit jobs. After the submitted job finishes, the dynamic system management server collects the information from other distributed computers. The dynamic management server and its assistant server move dynamically to new servers, if the present servers become busy. The dynamic system management server also demands the execution server to transfer the result data to the optimal data server. The dynamic system management server copies the computing data and sends the compressed computing data to another optimal data server in order for a robust data storage system. The clients can deploy their programs, execute jobs and retrieve the result data by accessing only the web page in the job execution service system. This job execution management server also has a function of automatic system construction, so that the users can manage the setup of the job execution management system easily on their closed distributed computers.

Another remarkable example of PSE is an education support PSE[32]. Network-based learning has been taking an important role in education as helpful education tools. However, it is difficult for teachers to retrieve education data from students or to obtain data from the student activities. Therefore, a problem solving environment (PSE) for the education and learning support: TSUNA-TASTE[32] was developed. The TSUNA-TASTE system collects the system-usage statistics, the information for the windows used and the operation situation of the mouse and key board of all students. The data, which the system TSUNA-TASTE collects, are stored in a database on the TSUNA-TASTE system server. Based on the data collected, teachers can have the learning status data for each student, and can guide the students in a better way.

Another research issue in PSE is validation, verification and uncertainty control in scientific simulations. When a software gives incorrect results for users, it may cause some difficulties, errors and accidents, depending on target problems[21), (33)~(37)]. The validation and verification mechanism is essentially important in scientific computing. This point was also pointed out by E. Houstis, J. Rice and his colleagues[38]. Standardization and benchmark problems in each field may help to perform the validation and verification. In addition, uncertainty management must be addressed intensively in order to avoid serious accidents and disasters in our society. PSE is one of candidates to manage the uncertainty in a relatively easy way[39]. The topic on the uncertainty is also discussed in this paper.

There are many PSE examples studied so far. In the references of (17)~(20) and (25) one can also find the example PSEs. In the next section some typical example PSEs are introduced.

## 3. PSE Examples

### 3.1 A Program Generation Support PSE

PSE studies [2-14] for partial differential equation (PDE) based problems have been extensively explored in order to support engineers and scientists to compute or simulate their problems and products on computers in e-Sciences and e-Productions. One of the major objectives in PSE researches is to help users compute or simulate their problems without heavy tasks, for example, without complete knowledge [11, 12] for computations or without a full programming [2-13].

In this subsection a program generation support PSE, called NCAS is presented. NCAS inputs partial differential equations (PDEs), the initial and boundary conditions, the discretization method and the algorithm, and outputs a C language program for the problem. The PDEs are automatically discretized and the program is generated mechanically. NCAS is a white box system, in which users can see and steer all the processes of the program generation. NCAS contains all the information for program generation, including the basic equations, the discretization schemes, the discretized equations, the boundaries and the initial conditions, the mesh structure, the program structure, and the definitions of all the variables and the constants. Therefore, a document for the corresponding program is also generated together with the program itself. In PSE for PDEs problems, one of problems, which should be addressed, is to develop huge PSE systems, including reusability of legacy PSE software. In order to solve this problem, a module-based PSE is proposed[9), (40)]; each PSE module solves a part of PSE tasks, for example, a problem description interface, a discretization module, a scheme suggestion module, a program flow designer, a program generator, a data analyzer, a visualizer, and so on. If each module can be developed independently and works cooperatively and smoothly to solve one PSE job, the heavy work of PSE development may be drastically relaxed. In this subsection a distributed PSE, called NCAS, is introduced, which supports users to generate computer programs[1)~(4), (9), (26), (40)].

The PSE system of NCAS inputs a problem information including discretization and computation schemes, and outputs a program flow graph, a C language source code for the problem and also a document for the program and for the problem. On a host computer a user inputs his/her problem, and the host guides the user to solve the problem. The distributed PSE for PDEs consists of several modules: a problem description, a

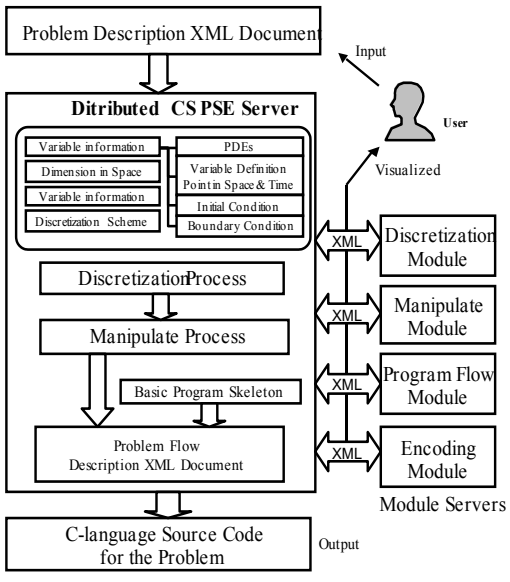


Fig. 2 Distributed-PSE: NCAS workflow.

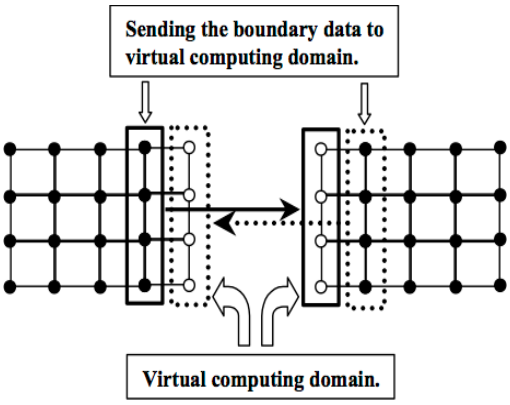


Fig. 3 Boundary data are communicated by the MPI functions in NCAS among domains decomposed. The MPI functions are automatically inserted into the program designed and generated in NCAS. The Finite Difference method (FDM) is employed in this example.

discretization, an equation manipulation, a program design, a program generation, documentation support, a module liaison and a job execution service. Each module is distributed on distributed computers, and all the information is described by the Extensible Markup Language (XML) including the Mathematical Markup Language (MathML). Each distributed module communicates with the host module by using XML documents, so that outputs from each module are visualized. Independent modules, which are developed by other engineers or users for one of the functions specified above can be also used after adjustments to the distributed PSE interface, if necessary. Therefore, the concept of the distributed PSE extends the potential of conventional PSE systems. The PSE contains all the information of the problem, PDEs, discretization scheme, mesh information, equation manipulation results, program design structure, variables and constant definitions and program itself. Therefore, the documentation support module also generates a document for the program generated and the problem itself in the PSE. The module liaison module generates an adapter module for the distributed PSE modules. The adapter module generated by the module liaison system inputs output data from preceding modules and/or external modules, and connects the data to the input data for the next module. The program generation PSE module provides a workflow shown in Fig. 2, and the user follows the workflow navigation for a problem generation.

The NCAS modules also help users generate MPI based parallel simulation programs based on partial differential equations (PDEs). The NCAS capability explores possibilities to visualize and steer the parallel program design process. At present NCAS supports a domain decomposition in a design of a parallel numerical simulation program, and the domain decomposition is designed or steered by users through a visualization window. After designing the domain decomposition, the parallel program itself is also designed and generated in NCAS, and the designed parallel program is visualized and steered by a PAD diagram. In NCAS, MPI functions are employed for message passing, and a SPMD (single program multiple data) model is supported. The visualization and steering capabilities provide users a flexible design possibility of parallel programming.

In the parallel program generation support in NCAS, for the data communication among the processors, the MPI functions are used. At least the boundary data for each domain decomposed are required to complete the computation in the adjacent processor (see Fig. 3). In NCAS the MPI functions are also automatically inserted to complete the parallel data communication programming. After specifying the domain decomposition information in NCAS, the parallel program is generated and provided to the users.

Figure 4 presents an example description of an input problem information, and Fig. 5 shows an example domain decomposition information. Through the NCAS visualization windows, for examples, shown in Figs. 4 and 5, one can check all the information and can also edit the information. In NCAS, after setting all the information for the problem description, the dicretization information and the parallelization information through the NCAS windows, all the information is visualized to

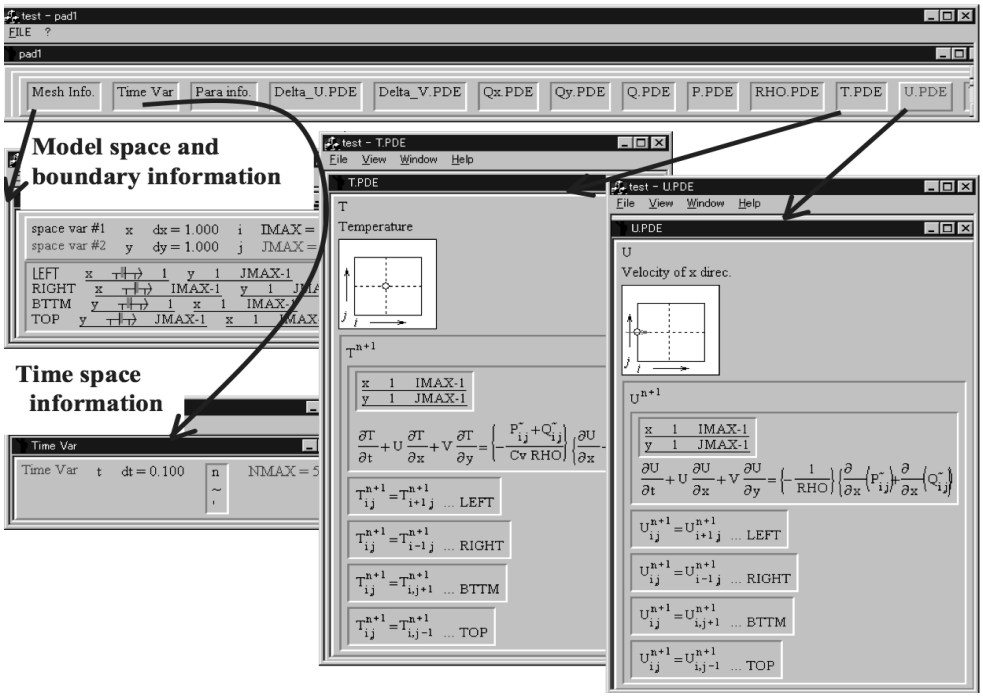


Fig. 4 An example PDE-problem description in NCAS. On each window users can edit the input description.

the users and the users can edit all the information through the windows. The discretization of each PDE is also performed automatically; depending on the discretization information which users input through the NCAS windows, the PDEs are discretized and manipulated appropriately according to the PDEs solving scheme. Then NCAS designs the parallel program for the problem, and outputs the parallel program and the corresponding document. Figure 6 shows an example MPI program automatically generated in NCAS.

In order to check the dynamic load balance function automatically generated by NCAS, during the computation an additional load was applied as shown in Fig. 7 (the left graph): by the additional load the computation time increases much in this specific case, if the static load balancing is used. When the dynamic load balancing method is selected in this example case, NCAS generates the functions, which measure the load balance of each machine dynamically, and according to the measured result each domain size is changed and adjusted dynamically to minimize the computation time. The right graph in Fig. 7 demonstrates the viability of the dynamic load balancing functions generated in NCAS, and the computation time reduction is significant in this case.

In the distributed PSE all the modules are distributed on network-linked computers. The information for the distributed modules and the computers are registered in a host computer. Newly developed modules by some users or scientists or so can be also registered in the host PSE server. The distributed PSE host server has the registered information for the modules oriented to one specific purpose, and users can obtain the information for each module and can select one of the modules to perform one task in all the PSE process.

The communication is accomplished through an interface using WWW server and Applet. The PSE server sends information described by XML to a module, and the module performs the task. The module sends the result based on the input XML information back to the PSE server. The result is visualized so that the user can check if the result is appropriate. After the successive processes, finally the NCAS generates a designed program flow and then a C program.

### 3.2 An Education Support PSE

Network-based e-Learning is one of important education ways. In addition, the network-connected personal computers have become popular to schools and homes widely. In the network-based e-Learning, each learner can access education

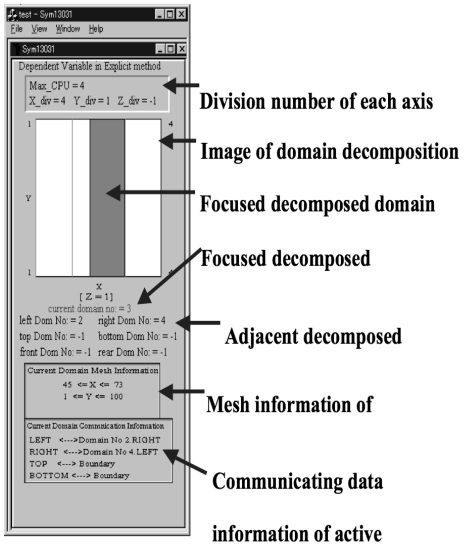


Fig. 5 Input and visualization of domain decomposition information.

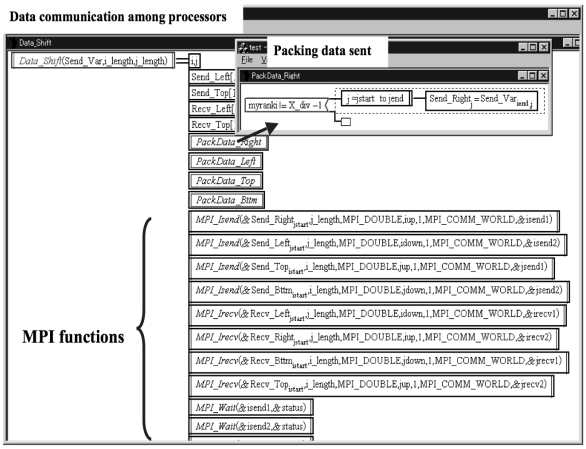


Fig. 6 Visualization of MPI functions designed for a domain decomposition information in NCAS.

contents through the network anytime and anywhere. In an actual education, a network-based e-Learning system becomes popular in a long-distant learning and at the same time in a class-room education. Even inside class rooms, each computer is connected and can be used as a detecting device for the learners' progress and status. An e-Learning server may have facilities such as a user identification method or a file sharing tool in the network-based environment.

It would be difficult for teachers to know the learning state of students through each personal computer connected by a network. Without the detailed information of the students' achievement, it is difficult for the teachers to perform an appropriate guidance and education depending on the students' learning level. Therefore, the state of the students is important and required for the suitable guidance. The education-support PSE system, which provides teachers the student-achievement information, helps them in their teaching planning or the appropriate guidance.

The PSE concept was proposed to support the programming activity and also to provide integrated human-friendly computational facilities in e-Sciences and e-Productions[(1)~(19)]. In this subsection, an education and learning support PSE system: TSUNA-TASTE is introduced[(32)].

The network education support system (TSUNA - TASTE) consists of four parts (see Fig. 8). The first part is an agent of student (Fig. 8(a)). It is a software, which always works

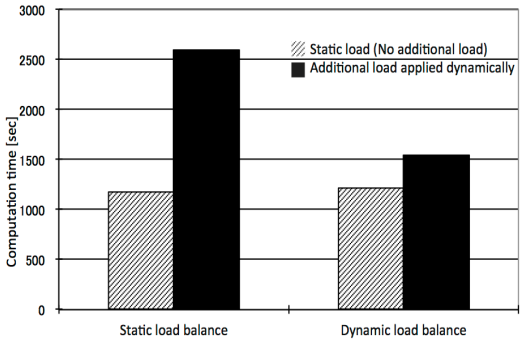


Fig. 7 A performance test result for a dynamic load balance. An additional load was applied during the computation, so that the computation time increases as shown in the left graph. When the dynamic load balancing function generated automatically by NCAS is used to this specific case, the domain size changes automatically depending on the machine load during the computation, and the result of the dynamic load balancing in the right graph shows the viability of the dynamic load balance functions generated by NCAS.

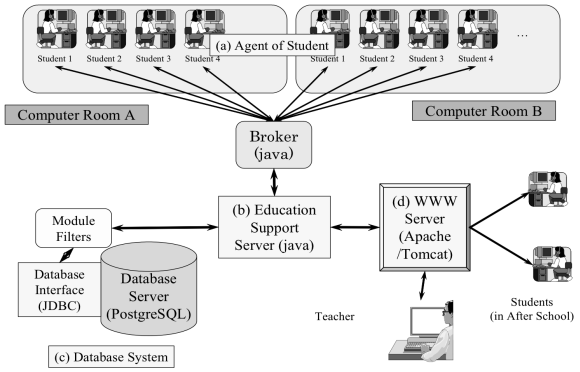


Fig. 8 Outline of the Computer assisted education support PSE: TSUNA-TASTE system.

on each personal computer of the student. The agent obtains the operation information of each student. The data are obtained from the operation information of the student through the OS with a resident software working on the personal computer of each student. The second part (Fig. 8(b)) is the education support server, that collects the data, which each student agent obtains via the network. The third part (Fig. 8(c)) is the database system. The database system stores the student profile data, the student personal data, the curriculum data and the teacher's personal data. The fourth part (Fig. 8(d)) is the web server displaying the information stored in the database. The web server (Servlet system) provides an interface to the TSUNA-TASTE handling.

The agent for each student resides on the memory of the personal computer of the student and performs the following three operations. Firstly the agent exchanges the messages with the education support server. The education support server transmits the messages to each agent. The student agent analyzes the messages, and obtains the process priority and the start time of the process described in the messages. The agent stores the message data in its task table. Secondly the agent manages the module program execution based on the task table. The module programs are small-size programs, which retrieve and output the student personal data from the student computers. The student personal data includes the achievement data, the operation data, the active window names and the process names. The third operation is a job to send back the data, which each module program collects, to the education support server. The module programs obtain the information of the student personal data from the OS of the student PCs. The module programs are implemented in the C++ language.

The module collects information about the student, however it does not store the raw data for security. It converts the raw data into statistics data. The transmission data are encoded and transferred. Furthermore, the personal information is not included in the data transmitted to the server. Thus, this system is robust for the electronic eavesdropping of data.

The education support server receives the operation data of the students through the student agent, and transmits the teacher's instructions to the students through the student agent. The education support server marks the students' absence, and identifies the students and their PCs. The education support server sends the messages of the data process demand to the agent of the student, and transmits the student personal data to the database server. The education support server also retrieves the student personal data requested by the teachers through the WWW server, and sends them back to the WWW server (see Fig. 8(d)) in the TSUNA-TASTE. The education support server is built using the Java language.

The data, which the education support server receives, are stored in the database. The database includes the private information of each student and teacher. The data contains the private information such as college student registration numbers, mail addresses and so on. These unchanged data are stored in the database together with the temporal data like the site of the student PCs. The server can receive the data from about 120 personal computers at the same time.

The web server system provides the user interface of the TSUNA-TASTE system. The teachers can obtain the state data of the students from the web system. The web server system presents the student achievement data, the learning progress and error occurrences situation during the programming exercises. The web system also provides an input interface to control the action of the TSUNA-TASTE: Through the web system, teachers can send a data gathering command, monitor the students' present usage of applications, and kill the unnecessary application processes on the students' PCs. The TSUNA-TASTE may open a new helpful e-Learning world.

### 3.3 Job Execution Support PSE on GRID

It is difficult for users to submit jobs to distributed computers on Cloud /Grid and to retrieve calculation data from them in scientific computing. In this subsection, a robust job execution service system is introduced in a closed distributed computer system[28), (29), (41)]. The job execution service system consists of a dynamic system management servers, execution servers and data servers as shown in Fig. 9. The dynamic system management server is duplicated in order to keep the system robust, and has an assistant management server. The dynamic system management server has a function of the job execution system management, including software deployment, program compilation, job execution, job status retrieval and computing data retrieval. This system does not require special middleware for Cloud /Grid. Users access the web page on the dynamic system management server, and the clients submit jobs. After the submitted job finishes, the dynamic system management server collects the information from other distributed computers. When the present servers become busy, the dynamic management server and its assistant server move dynamically to new servers. The dynamic system management server also demands the execution server to transfer the result data to the optimal data server. The dynamic system management server copies the computing data and sends the compressed computing data to another optimal data server in order for a robust data storage system. The clients can deploy their programs, execute jobs and retrieve the result data by accessing only the web page

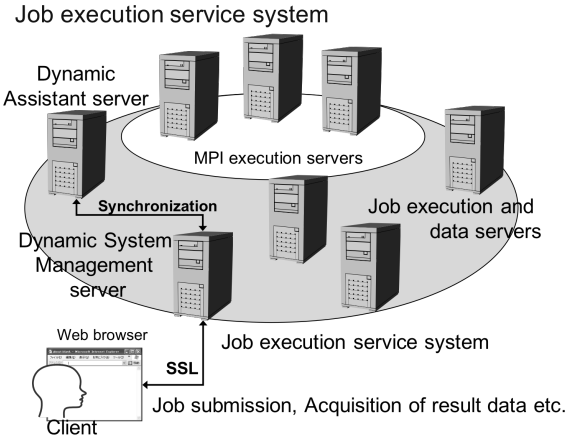


Fig. 9 Job execution service on distributed computers n Cloud / Grid.

in the job execution service system. This job execution management server also has a function of automatic system construction, so that the users can manage the setup of the job execution management system easily on their closed distributed computers.

Users access the web page on the dynamic system management server, and the clients submit jobs. After the submitted job finishes, the dynamic system management server collects the information from other distributed computers on Cloud / Grid. The clients can deploy their programs, execute jobs and retrieve the result data by accessing only the web page in the dynamic system management server.

The job execution service system acquires necessary resource information for servers for job execution and for data retrieval and storage. The resource information contains CPU architecture name, CPU operation frequency, total memory, memory in use, unused memory, load average and unused capacity of hard disk. The system sorts the resources in order for effective job execution. The users can find the resource information on the job execution service system Web page.

Clients access the dynamic system management server through the Web page of the system, and perform computing. Through the Web page, the clients can up-load source files or executables to the dynamic system management server, select computing servers from among resources recommended by the system, and set execution environment. When two or more files are required for one job, the client should compress those files. When MPI jobs are executed, computing servers, on which MPI is installed, are recommended to the clients.

The compilation command, the execution method, the comment and the server name for job execution are specified by clients. The clients can also specify the storage location of the result data of the job. When the clients do not especially specify the storage data server, the system forwards the result data to the best data server. When input information is not sufficient, the job execution service system displays an error message, and advises to input the required input data. The clients can select the execution methods, or make scripts for the execution on the Web page in a compressed file format.

When the job setting ends, the job execution service system forwards the job to a pertinent server or a server set based on the setting information. The setting file is described in XML, and contains the computing server information, the compilation command, the execution information and the data storage server information. When a compressed file, which contains source files/binaries and a make file, is sent to the computing servers, the compressed file is decompressed and the decompressed information is sent to the dynamic system management server, so that the clients can check if the file decompression is succeeded. When the compilation or the execution errors appear, the computing server notifies them to the dynamic management server. When the execution server specified is occupied by another job, the job is scheduled by the dynamic system management server.

When a job ends, the job execution server forwards the result data to a pertinent server based on the setting information. In addition, its compressed result data is stored in another data server. The result data duplication makes the data server robust and fault tolerant.

When no data server is specified in the setting information, the computing server asks the dynamic system management server about the data storage servers. Based on the unused hard disk capacity, the better two data storage servers are selected from among the servers, on which no jobs run. One is for the result data uncompressed and the other is for the compressed backup data. When the result data is stored on the data servers specified, the data server locations are notified to the dynamic system management server. The client can refer to and can download the data from the Web page on the dynamic system management server.

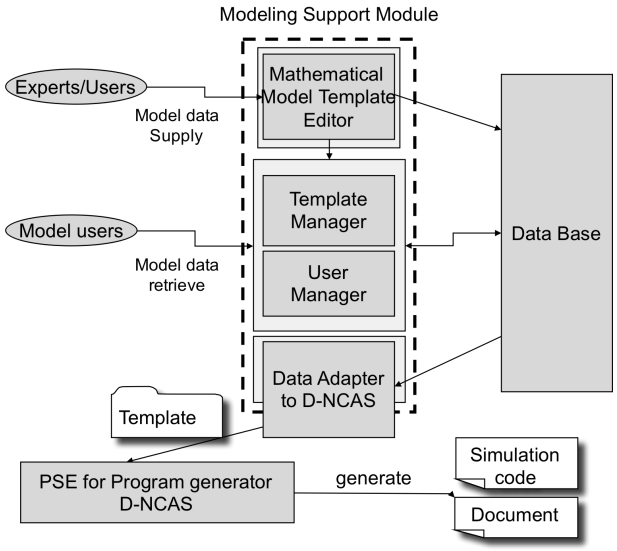


Fig. 10 Mathematical modeling support module in a PSE of NCAS for PDE-based problems (see subsection 3.1). The module has two functions: Experts input their knowledge through the Model template editor and it is stored in the data base. Model users retrieve the mathematical models from the database for their problems and modify them.

### 3.4 Mathematical Modeling Support PSE[42)~(45)]

In development of computer programs, mathematical models are necessary to obtain reasonable or correct results for the users' problems. PSE research also covers the mathematical modeling support[(42)~(45)]. If the users are not experts in the target scientific area, the mathematical modeling support is helpful to build up their mathematical models. Even for experts for the target areas the mathematical modeling brings them heavy tasks. The PSE function for the mathematical modeling may help the users to derive mathematical basic equations. Therefore, the mathematical model construction support module is required and should be studied.

On the other hand, each expert can supply mathematical models in scientific expert areas. If PSE systems support a database construction function for the mathematical models, each expert can input his/her knowledge to enrich the mathematical model database, on which afterward users utilize the data pool for the mathematical models to derive their own mathematical models.

A new module of mathematical model construction support is introduced, as an example, in a PSE of NCAS (see Fig. 10)[(45)]. The module has two important functions: 1) mathematical model database construction support and 2) mathematical model construction support. In the database construction, the mathematical model construction support module supplies templates for mathematical models: problem information, equations, key words and information for equations themselves, terms, and characters (symbols, constants and variables), in addition to the group and user permission. Experts in some scientific areas or users supply the information through the template, and the mathematical model database is enriched. The model suppliers can set the access authorization to the mathematical model database. Then the mathematical model support module supports users to have their mathematical models based on their input of key words for their target problems, and outputs the mathematical equations, for example, PDEs. After receiving the results for the mathematical models, the users can also modify them and use them to solve their problems. The mathematical model constructed is supplied to

NCAS (see subsection 3.1) to generate a C program. The model data stored in the mathematical model database is also described by XML. Through the mathematical editor, one can input equations easily, and the editor outputs the XML format data.

By the mathematical modeling support concept based on the model database construction support it is expected to enrich the mathematical model database. Therefore, the mathematical modeling support may contribute the PSE world extraordinarily.

## 4. Future of PSE

As we described above, computer hardware and algorithm capabilities have grown extraordinarily, while the programming power has been explored slowly. However, some part of programming task is mechanical, so that the part can be done by computers / software mechanically. If the PSE provides realistic services, we can devote ourselves to the target problems themselves.

The PSE is not a general purpose system nor a software but the concept described above and in the paper. Therefore, each PSE needs specialists for the target problems and also collaborations among PSE developers and the specialists. The PSE challenges include the following issues: The most important issue must be how to describe the problems. The issue should be addressed together with the specialist for the problems and PSE developers to make our e-Life, e-Science, e-Production, etc. rich. The target problems are diversified in various fields. PSEs cooperate with other PSEs, softwares, hardwares, sensors, network, robots and human activities in making our life comfortable and rich. In order to find out the target problems, collaborations among specialists and PSE researchers are essentially required. Human beings have discovered new problems to be solved, and the solutions have improved our life in many ways. Some of them have led to revolutions and new trends in our society. PSEs serve concepts and tools to solve the problems in the unique view of PSEs, as presented in the paper. The PSE systems tend to be huge integrated systems, which consist of softwares and hardwares. How to design and develop PSEs are other important issues. Specialists on the target problem, hardwares, programming and PSE should collaborate and communicate to design and develop an effective PSE. A modular way for the PSE design and development is one of solutions for the issues, so that the PSE can be easily developed and maintained. The PSE may have multi-functions; in this case each function may be developed as a separate module. One solution is a modular way to design and develop PSEs. Each module can be developed by independent developers. Each PSE and each module should be easy for maintenance to fit or include users' future further requests. For the module-based PSE, the interface between modules should be taken care and be matched each other. The input / output information must be clearly described and opened. A module-liaison module may also play an important role in this aspect. Some standardization may help to match the interface among modules.

Verification and validity of PSEs and PSE solutions are other challenges. Expected output should be reproduced for known input in PSEs in each field. In addition, each PSE needs many use cases by many users, and from the users PSE developers can get feedback to improve the PSE. The continuous cooperation among users, PSE developers and specialists in each field are essentially important to improve PSEs for real use in our world.

Enormous increases in the computer hardware power and the advances in algorithms have pushed up PSE research activities extensively. In the 1960s, right after the high-level language appearance, trials of PSE-like activities have started. However, at the time, computer technology was not matured enough to support the trials of computer-assisted environment. In these decades the PSE studies have been explored very much, as introduced in part in this paper. Most international and domestic conferences and meetings relating to computational sciences and engineering usually include PSE as one of important topics. Now it would be realistic to dream powerful serviceable PSEs in the near future. The PSE enriches our e-Life, e-Production, e-Science, etc.

One example may be found in our near-future home. When you decorate or makeover your rooms, you may go shopping to see interiors and painting color. Or you may change layout of beds, chairs, desks, etc. and to look around the real layout change. Sometimes you want to change a position of your air conditioner in a room. In this case it is not so easy to change the positions in a real room to check the air environment. In these cases PSEs would provide an easy and simple simulation tools. One can visualize the room layout on an interactive virtual visualization device to check the arrangement, before changing the real layout. The interactive visualization is being a daily common tool at home at present. The voice instruction and hand movement may reflect users' intentions and may act as real-time and interactive input methods.

In science and engineering, computer simulations may become a real tool for scientists and engineers to think target problems, and devote themselves to the target problems. At present, when we perform simulations for a new phenomenon, normally we have to develop a new computer program or to learn how to use a CAE software. To develop a program or to learn how to use the CAE software may take a long period, depending on the program complexity. A PSE would help researchers to enable discovery faster and engineers to develop a new product. Even now we should consume time and intensive efforts to do simulations, before reaching to discoveries and developments. PSEs may generate programs instead of human and may navigate users to use CAE software. A scientist may have a scientific idea. He or she may sit down on a sofa and talk with PSEs, which may present a mathematical model on a visualization device. Interactively the scientist may modify the model for the purpose. Then PSEs create a program, which may be run on a cloud environment including super computers[53] and visualize the results produced. The results may be analyzed by the users' voice commands. These user-friendly problem solving environments are almost realizable now. The near-future PSEs will communicate with people, sensors, machines, software, other PSEs and hardware distributed on a network / cloud to solve daily problems, scientific problems, and engineering problems. Our future life will be supported and surrounded by invisible / visible PSEs.

## 5. Conclusions

The PSE has been extensively explored over the past few decades. The explorations have been supported by the reinforced computer power and algorithm power. PSE will boost up the programming power, and enrich our e-Life and e-Science. In the near future of the PSE development we should consider how to create a PSE to help create other PSEs. To build up a PSE is a very hard task and needs huge human efforts. Therefore, PSE researchers have still been meeting this difficulty. In this research issue meta PSE or PSE for PSE may play an important role to build up service-oriented PSEs. One example of the meta PSE is a PSE Park[39], in which many modules, developed by PSE researchers / developers, are distributed. Each module has one function or may be a one PSE, and is developed by many independent researchers and developers. By connecting the modules, PSE researchers or users can construct a single-purpose PSE or so. The interface should be opened, so that each PSE connector can be easily developed by each user or researcher or developer. The module mediator / connector may come into play there. The module base PSE may open a new PSE World.

Another research issue in PSE is validation, validity and uncertainty. When a PSE gives a wrong result for users, it may cause some difficulties, errors and accidents, depending on target problems. The validation and verification mechanisms are essentially important as usual software. Standardization and benchmark problems in each field may help to solve the validation and verification. This issue is a general common problem in software.

**Acknowledgements**

This work was supported partly by JSPS, MEXT and the Japan Society of Computational Engineering and Science (JSCES). The authors would like to also extend their appreciations to Prof. J. Rice, Prof. E. Houstis, friends in IFIP WG2.5, and friends in the PSE research group in Japan including Prof. T. Teramoto, Dr. H. Usami, Dr. T. Maeda, Dr. Y. Manabe, Dr. H. Kobashi, and our former students and friends contributed to the works relating to the paper.

**References**

(1) Boonmee, C. and Kawata, S., Computer-Assisted Simulation Environment for Partial-Differential-Equation Problem: 2. Visualization and Steering of Problem Solving Process, *Trans. of the Japan Society for Computational Engineering and Science*, Paper No. 19980002, 1998.

(2) Boonmee, C., Kawata, S., Fujii, S., Manabe, Y. and Tago, Y., Visual Steering of the Simulation Process in a Scientific Numerical Simulation Environment - NCAS -, in print, *Enabling Technologies for Computational Science*, edited by E.Houstis and J.Rice, Kluwer Academic Pub., 2000.

(3) Fujio, H., & Doi, S. (1998). Finite Element Description System as a Mid-Layer of PSE. Proceedings of Conference on Computation Engineering and Science, 3(2), (pp. 441-444).

(4) Fujita, A., Teramoto, T., Nakamura, T., Boonmee, C. and Kawata, S. (2000). Computer-Assisted Parallel Program Generation System P-NCAS from Mathematical Model-Visualization and Steering of Parallel Program Generation Process-. Transaction of the Japan Society for Computational Engineering and Science, Paper No. 20000037.

(5) Gallopoulos, E., Houstis, E. and Rice, J.R., Future Research Directions in Problem Solving Environments for Computational Science, Technical Report CSRD Report No. 1259, *Report of a workshop on Research Direction in Integrating Numerical Analysis, Symbolic Computer, Computational Geometry, and Artificial Intelligence for Computational Science*, Washington DC, April 11-12, 1991.

(6) Gallopoulos, E., Houstis, E. and Rice, J. R. (1994). Computer as Thinker/Doer: Problem-Solving Environments for Computational Science. IEEE Computational Science and Engineering, 1(2), 1-23.

(7) Hirayama, Y., Ishida, J., Ota, T., Igai, M., Kubo, S. and Yamaga, S. (1988). Physical Simulation using Numerical Simulation Tool PSILAB, Proc. *1st Problem Solving Environment Workshop*, pp. 1-7.

(8) Houstis, E. N. and Rice, J. R., Parallel ELLPACK, a development environment and problem solving environment for high performance computing machienes, edited by In P. Gaffney and E. N. Houstis, *Programming Environments for Hihg-Level Scientific Problem Solving*, North-Holland, Amsterdam, 1992, pp. 229-241.

(9) Kawata, S., Boonmee, C., Fujita, A., Nakamura, T., Teramoto, T., Hayase, Y., Manabe, Y., Tago, Y. and Matsumoto, M., *Visual Steering of the Simulation Process in a Scientific Numerical Simulation Environment –NCAS-*, Enabling Technologies for Computational Science, E. N. Houstis and J. Rice (Ed.), Kluwer, 2000, pp. 291-300.

(10) Okochi, T., Konno, C. and Igai, M., High Level Numerical Simulation Language DEQSOL for Parallel Computers. *Trans. of Information Processing Society of Japan*, 35(6), 1994, 977-985.

(11) Umetani, Y., DEQSOL A numerical Simulation Language for Vector/Parallel Processors, Proc. *IFIP TC2/WG22*, 5, 1985, pp. 147-164.

(12) Rice, J. R. and Boisvert, R. F., Springer Series in Computational Mathematics 2, *Solving Elliptic Problems Using ELLPACK*, Springer-Verlag, New York, 1984.

(13) Kawata,. S., Computer Assisted Problem Solving Environment (PSE), *Encyclopedia of Information Science and Technology, Third Edition*, ed. Mehdi Khosrow-Pour, Chapter 119, 2014, pp. 1251-1260, IGI Global, Hershey, PA, USA.

(14) Kawata, S., Computer Assisted Parallel Program Generation, *Encyclopedia of Information Science and Technology, Fourth Edition*, ed. Mehdi Khosrow-Pour , Chapter 398, 2017, IGI Global, Hershey, PA, USA.

(15) Kawata, S. Review of PSE (Problem Solving Environment) Study, *J. Convergence Information Tech.*, 5 2010, pp. 204-215.

(16) IFIP WG2.5 (International Federation for Information Processing, Working Group 2.5), IFIP WG2.5 homepage retrieved on June 12, 2017, http://www.ifip.org/wg-2.5

(17) Ford, B. and Chatelin, F. (Ed.), *Problem Solving Environments for Scientific Computing*, North-Holland, 1987.

(18) Gaffney, P. W., and Houstis, E. N. (Ed.), *Programming Environments for High-Level Scientific Problem Solving*, North-Holland, 1992.

(19) Houstis, E. N., Rice, J. R., Gallopoulos, E. and Bramley, R. (Ed.), *Enabling Technologies for Computational Science, Framework, Middleware and Environments*, Kluwer Academic Publishers, 2000.

(20) Gaffney, P. W. and Pool, J. C. T. (Ed.), *Grid-Based Problem Solving Environments*, Springer, 2007.

(21) Dienstfrey, A. and Boisvert, R. (Ed.), *Uncertainty Quantification in Scientific Computing*, Springer, 2012.

(22) JSCES (The Japan Society for Computational Engineering and Science), JSCES homepage retrieved on June 12, 2017, http://www.jsces.org/.

(23) PSE Research Group in Japan, homepage retrieved on June 12, 2017, http://www.jsces.org/activity/research/pse/.

(24) Rice, J. R. and Boisvert, R. F., *Solving Elliptic Problems Using ELLPACK*, Springer Series in Computational Mathematics 2, Springer, 1984.

(25) Kawata., S., Tago, Y., Umetani, Y. and Minami, K. (Ed.), *PSE Book: Computer assisted Problem Solving Environment in computing science (Basic & Advanced) (in Japanese)*, Tokyo: Baifukan, 2005.

(26) Inaba, M., Fuju, H., Kitamuki, R., Kawata, S. and Kikuchi, T., Computer-Assisted Documentation in a Problem Solving Environment (PSE) for Partial Differential Equation Based Problems, *Trans. of the Japan Society for Computational Engineering and Science*, 20040025, 2004.

(27) Message Passing Interface Forum, MPI: A Message-Passing Interface Standard, *Technical Report*, University of Tennessee, Knoxville, Tennessee, 1994.

(28) Fuju, H., Kawata, S., Sugiura, H., Saitoh, Y., Usami, H., Yamada, M., Miyahara, Y., Kikuchi, T., Kanazawa, H. and Hayase, Y., Scientific Simulation Execution Support on a Closed Distributed Computer Environment, *2nd IEEE International Conference on e-Science and Grid Computing*, 2006, 27340112.

(29) Kanazawa, H., Itou, Y., Yamada, M., Miyahara, Y., Hayase, Y., Kawata, S. and Usami, H., Design and

Implementation of NAREGI Problem Solving Environment for Large-Scale Science Grid, *2nd IEEE International Conference on e-Science and Grid Computing*, 2006, 27340105.
(30) Globus, Globus Online. Retrieved June 12, 2017, http://www.globus.org/
(31) UNICORE, UNICORE. Retrieved June, 2017, http://www.unicore.eu/
(32) Teramoto, T., Okada, T., and Kawata, S., A Distributed Education-Support PSE System, *3rd IEEE International Conference on e-Science and Grid Computing*, 2007, pp. 516-520.
(33) CMFVVUQ (Committee on Mathematical Foundations of Verification, Validation, and Uncertainty Quantification), *Assessing the Reliability of Complex Models: Mathematical and Statistical Foundations of Verification, Validation, and Uncertainty Quantification*, The National Academies Press, 2012.
(34) Johnson, C. R., Top Scientific Visualization Research Problems. *IEEE Computer Graphics and Applications: Visualization Viewpoints*, 24(4), 2004, pp. 13-17.
(35) Kahan, W., Desparately Needed remedies for the Undebuggability of Large Floating-point Computations in Science and Engineering, *paper presented at IFIP Working Conference on Uncertainty Quantification in Scientific Computing*, retrieved on June 12, 2017, http://www.eecs.berkeley.edu/~wkahan/Boulder.pdf
(36) Rump, S. M. (2010). Verification methods: Rigorous results using floating point arithmetic. Acta Numerica, 19, 287-449.
(37) Weimer, W. R., *Exceptional Situation and Program Reliability*, PhD Thesis, University of California, Berkeley, 2005.
(38) Houstis, E. N., Gallopoulos, E., Bramley, E. and Rice, J. R., Problem-Solving Environment in Computational Science. *IEEE Computational Science & Engineering*, 4, 1997, pp. 18-21.
(39) Kawata, S., Kobashi, H., Ishihara, T., Manabe, Y., Matsumoto, M., Barada, D., Hayase, Y., Teramoto, T. and Usami, H., *Scientific Simulation Support Meta-System: PSE Park - with Uncertainty Feature Information -*, International Journal of Intelligent Information Processing, 3, 2012, 66-76.
(40) Teramoto T., Nakamura, T., Kawata, S., Matide, S., Hayasaka, K., Nonaka, H., Sasaki, E. and Sanada, Y., A Distributed Problem Solving Environment (PSE) for Partial Differential Equation Based Problems, *Trans. Jpn. Soc. Comp. Sci. Eng.*, Paper No. 20010018, 2001.
(41) Kawata, S., Usami, H., Hayase, Y., Miyahara, Y., Yamada, M., Fujisaki, M., Numata, Y., Nakamura, S., Ohi, N., Matsumoto, M., Teramoto, T., Inaba, M., Kitamuki, R., Fuju, H., Senda, Y., Tago, Y. and Umetani, Y., A Problem Solving Environment (PSE) for Distributed Computing, *Int. J. High Performance Computing and Network*, Vol. 1, No.4, 2004, pp. 223-230.
(42) Joshi, A., Houstis, C. E., Houstis, E. N., Rice, J. R. and Weerawarana, S., PYTHIA: a knowledge-based system to select scientific algorithms, *ACM Trans. on Mathematical Software*, 1988, pp. 447-468.
(43) Catlin, A. C., Houstis, C. E., Houstis, E. N., Rice, J. R., Ramakrishnan, N., Verykios, V. S., PYTHIA-II: a knowledge/database system for managing performance data and recommending scientific software, *ACM Trans. on Mathematical Software*, 1988, pp. 227-253.
(44) Finn, D. P., Hurley, N. J. and Sagawa, N., AI-DEQSOL: A knowledge-based environment for numerical simulation of engineering problems described by partial differential equations, *Artificial intelligence for engineering design, analysis and manufacturing*, 1992, pp. 199-212.
(45) Kawata, S., Teramoto, T., Sugiura, H., Saitoh, Y. and Hayase, Y. Mathematical Modeling Support in a Distributed Problem Solving Environment for Scientific Computing, *2nd IEEE International Conference on e-Science and Grid Computing*, 2006, 27340101

# Data reduction by software point rendering

Hideo Miyachi

Tokyo City University, 224-8551 3-3-1 Ushikubo-nishi Tsuzuki-ku Yokohama City, Kanagawa

I have developed a data reduction system with rendering process, which converts surface data to point data according to the resolution of the viewer window. Here, I have applied the method to extract a subset of point data. To keep the coordinate value of point data, I have implemented the system without graphics Z buffer. As the results of conducting preliminary tests, I learned that it worked well for data reduction and the coordinate value was kept after extraction. The introduction of ID buffer has made the overlapping point elimination effective.

***Key Words :*** *Visualization, software rendering, data reduction*

## 1. Introduction

The data size of scientific visualization has been growing year by year. It has been bringing the problem of large data visualization. The problem happens not only in the super computing field, but also, observation and measurement field. I have been aiming visualization solution by using point data that is replaced as alternative of polygon data [1.2]. The motivation comes from the difference as shown in Fig.1. In 1990's, the number of data outputted from simulation program was coarse against the resolution of the display. Recently, however, a number of polygons or nodes with simulation data are included in a pixel in the case of large scale simulation. In such situation, the interpolation is a waste work.

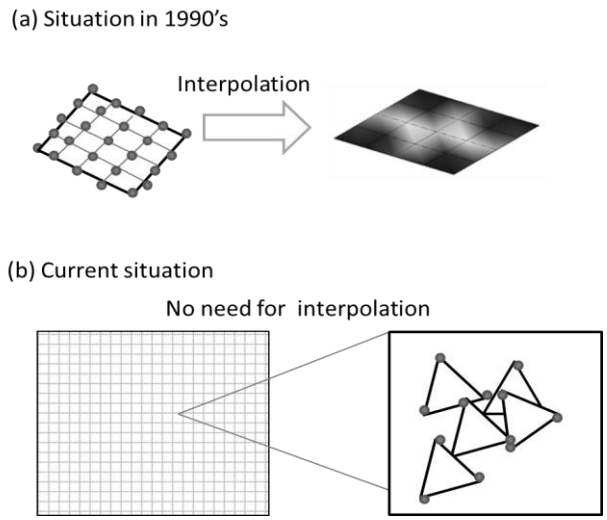


Fig.1 Change of the relationship between data nodes and pixels in scientific visualization.

To avoid the waste work, I have developed a data reduction system which converts surface data to point data by using OpenGL rendering implemented on graphics board [3]. In this study, I applied the method to point data which will be imported through measurement devices. The latest system implemented original software rendering for dedicating point rendering. Because it can keep the original coordinate value and can be applied to huge volume of point data not to be restricted with the graphics memory capacity. This paper describes the application example and some discussion on the method.

## 2. Data reduction by using rendering process

This method was developed for creating a lightweight model as an AR contents from CAD data. AR contents must be updated at an interactive speed over at least 30 frames per second. To realize the speed, it should be light. However, the full model defined by CAD become recently large, the polygon data converted to AR system cannot be handled with AR software. And the conventional data reduction method could not reduce them to enough weight models because the reduction ratio depends on the complexity of the original model. On the other hand, AR content has an advantage that the viewpoint for AR is limited. As shown in Fig.2, AR content must be displayed with AR marker on the display. Trying zooming up the car, the system may not recognize the AR marker from the partial AR marker in the camera view. The limitation of the viewpoint is not only the distance, but also the direction. The car will be never viewed from the bottom side, because the car is designed to be displayed on the AR marker.

Fig.2. A car displayed on a AR marker

Under such conditions that the area of viewpoint is extremely restricted, the alternative points can express the same quality of the image obtained by the surface rendering. Figure 3 illustrates the method to generate the alternative points. First, the viewpoints and directions in the field from where the model will be viewed are defined. Second, multiple images are rendered with the specified view information. The color information and the depth information were obtained from the frame buffer and the Z buffer respectively at each view. Third, all of the information was merged. It represents a set of 3D point cloud. Finally, the 3D point data can be displayed by OpenGL point viewer we developed. The 3D point cloud can be seen as the enough quality as the image from the surface rendering. A result of the comparison between the surface and the alternative points is shown in Fig.4. The average PSNR of (1) and (2) in 27 views was about 67.0 Db. It is the value that human cannot understand the image degradation. The difference caused the object edge area. Because a point object in OpenGL always occupies one pixel On the other hand, the OpenGL triangle object may not be displayed if its size is too small.

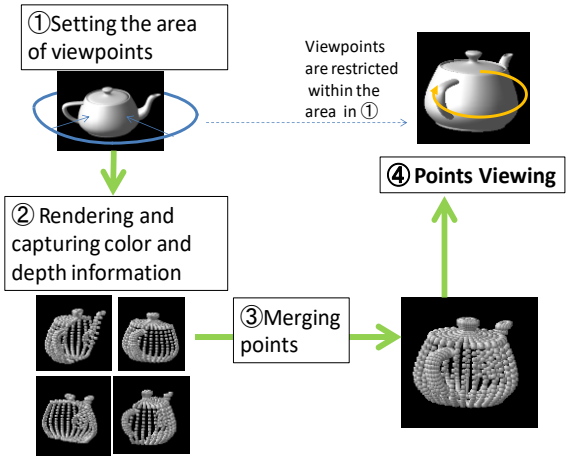


Fig.3 Alternative points generation process by using the rendered images from multi-directions.

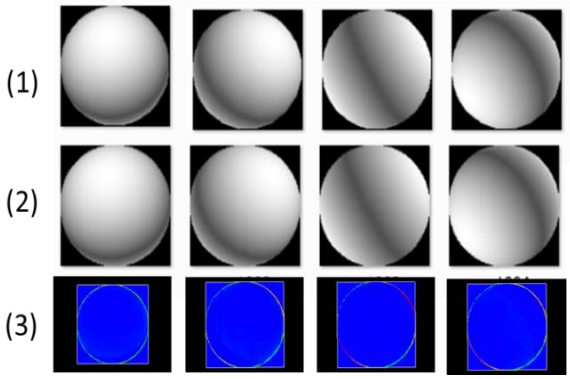


Fig.4 Quality comparison. (1) rendering images of a sphere represented surfaces (Triangles) , (2) rendering images of the alternative points, (3) The difference between (1) and (2).

## 3. Point data reduction

Large data problem occurs on not only supercomputing field, but also measurement field. It is not difficult that more than billion 3D point clouds are generated from laser and/or photographic measurement. Theoretically, the method introduced in the chapter 2 can be applied to point clouds, but two additional challenges were conducted. All of point data are the fact that is measured. Therefore, the data degradation using Z buffer should be avoided. An ID buffer that works as same as Z buffer but stores ID of the point instead of the depth was implemented. At the finishing the rendering, the pixels with ID that remains in ID buffer are visible pixel. To implement the ID buffer, and also to handle large data, the rendering program was implemented with main memory and CPU. The purpose of the rendering is not to create beautiful image but extract visible pixels. The renderer that dedicated only to extract visible points can perform enough speed without GPU.

## 4. Benchmark test

The prototype system was verified by two kinds of benchmark tests. The first one is to check the data reduction ratio. The second one is to check the overlapping points.

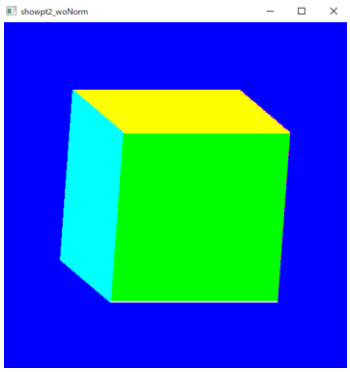

Fig.5 test data consists of 1,500,000 points on a cube shape

The visualized image of test data is shown in Fig.5. It looks a cube, but it does not have any surface. It consists of 1,500,000 points. A surface consists of 500 x 500 points.

Performing the data reduction, two kinds of parameter were applied. The first was the number of viewpoints and the second was the resolution of the rendering. Regarding with the number of viewpoints, three kinds of parameter, 2, 6 and 8 directions were applied as shown in Fig.6. The parameter W which indicates the resolution of frame/ID buffer was applied the three kinds of numbers, 150, 300 and 600. Here, W=300 means that the frame buffer has 300x300 pixels resolution. The scale of the object, now it is Cube, was normalized to the display region. The length of the cube diagonal is $\sqrt{3}$. The width and the height of the viewport was adjusted to the size, $\sqrt{3}$ as shown in Fig.7.

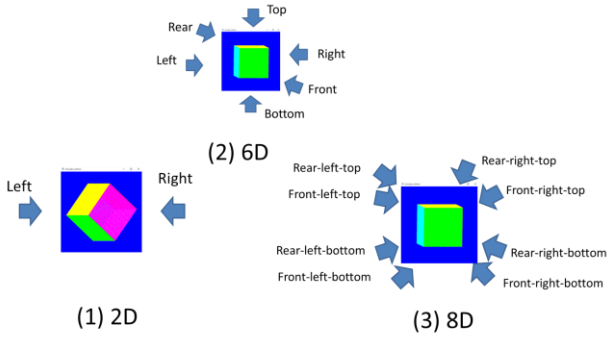


Fig.6 Three kinds of viewpoint parameter sets,(1) 2 directions in diagonal , (2) 6 frontal directions of 6 surfaces , (3) 8 diagonal directions of 8 vertices.

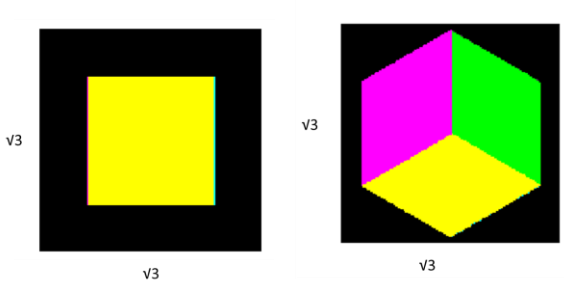


Fig.7 Object scaling

At the combination of the three kinds of viewpoints and W respectively, the data reduction was performed. The results are shown in table 1.

Table 1 Data reduction performance

| W/ Directions | オリジナル | Parameter W | | |
|---|---|---|---|---|
| | | 600 | 300 | 150 |
| D2 | 58.0 | 16.1 | 4.0 | 1.0 |
| D6 | 58.0 | 27.8 | 7.0 | 1.7 |
| D8 | 58.0 | 33.2 | 12.7 | 3.4 |
| | | | | [MByte] |

The data size after the reduction is depends on the sum of the projection area. The logical foreground area is 65% for D 2 and D 8, 33% for D 6. D2 and D3 do not have overlapping points except the edge area. The meaning of overlapping is shown in Fig.8.

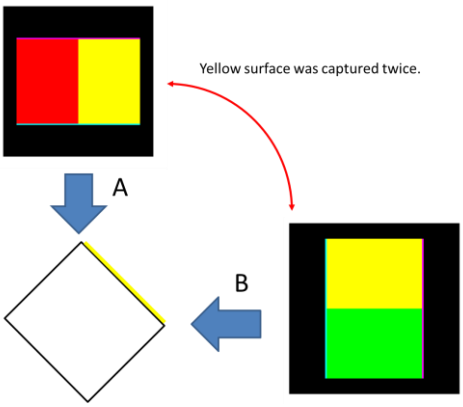


Fig.8 Overlapping: the images captured from the diagonal viewpoint include the same surface twice or more.

Theoretical area of the frontal of D2(W=600) is 468,000 points. It should be 31.2% of the original points. However, the actual data size is reduced to 27.8%. The ratio of actual/theoretical is 88.9%. One of the reason is that the overlapping points were eliminated. In the cases of W=300 and 150, the ratio is 88.4, it is almost the same as the ratio of W=600.

Theoretical area of the frontal of D6(W=600) is 712,800 points. It should be 47.5% of the original points. The actual data size was reduced to 47.9%, the ratio of actual/theoretical was 100.9%. In the cases of W=300 and 150, the ratio was 101.6% and 98.7% respectively. The same ratio of D8 was 45.9%, 70.2%, 75.2% at W=600,300 and 150 respectively. These ratios are plotted in a graph shown in Fig.9. The theoretical value does not include the overlapping point elimination. Therefore, the value of D8(W=600) is 1,872,000 (125.8%), the data size is expanded from the original. However, actually the overlapping data elimination reduced the data size to about 46% of the theoretical value.

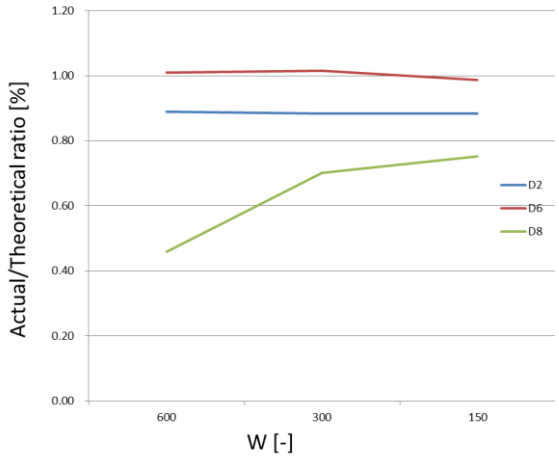


Fig.9 Actual/theoretical ratio of the data reduction

In the previous study, the elimination of overlapping points was not effective, so the increase in the number of viewpoints was directly related to the increase in the amount of data. However, in this study, introducing ID buffer has made the overlapping point elimination effective.

Finally, the visualized images of nine kinds of the reduced data are shown in Fig.10. No noise due to the data reduction is recognized in Fig.10 (1)-(3). A few noises can be seen in Fig.10 (4)-(6). And in Fig.10 (7)-(9) the fact that the cube consists of points is recognized obviously.

## 5. Conclusion

I have developed a new version of the data reduction system based on point generation which does not Z buffer on graphics board but on main memory. The implementation has brought

extensibility to the system, enabled to introduce ID buffer to avoid the data degradation due to the discretizing the depth value. Furthermore, it enabled the overlapping point elimination. As the results, even when setting an excessive viewpoint, data reduction works effective. In the future, I would like to apply this method to more complicated models.

Acknowledgements: This work was partially supported by Grants-in-Aid for Scientific Research No.17K00162 from Japan Society for the promotion of Science. I would like to express my gratitude to Mr. Matsumoto (CYBERNET SYSTEM Co. LTD.) who gave me his contribution of the software implementation and his insightful comments.

References

[1] Miyachi H , Sakamoto N: Data Reduction by Applying Image-Based Modeling and Rendering Technique to CG models, Journal of Visualization, Vol. 8, No. 4, pp. 331-338, 2005

[2] Hideo Miyachi: A Study of Data Reduction for Intuitive AR Interface, Proc. of ICAST2013(International Joint Conference on Awareness Science and Technology), pp. 328-334, 2013

[3] Hideo Miyachi: Quality evaluation of 3D image represented by points, AROB2016, pp. 686-689, 2016

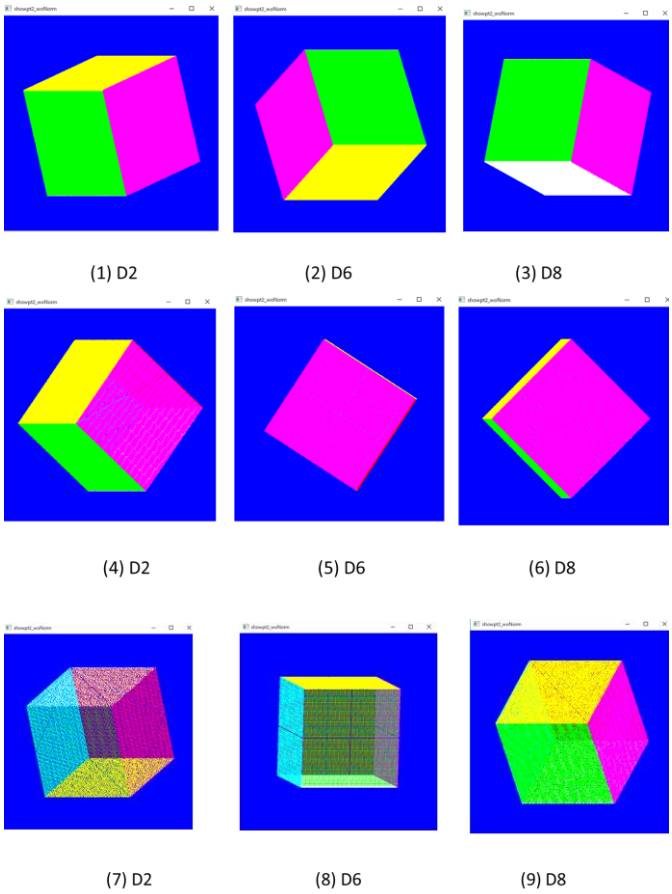


Fig.10 Visualized image of the reduction data in 500x500 pixels window by the point viewer with OpenGL. (1)-(3) W=600, (4)-(6) W=300, (7)-(9) W=150.

# Dixiter: A card game AI of Dixit

Hiro Kobashi[1)]
1) Fujitsu Laboratories of Europe, Hayes, UK

Abstract Dixiter is a computer game player for Dixit card game. The card game is very unique comparing with other typical card game like Porker, Bridge and so on because a player cannot be neither too strong nor too weak. This *so-so* behavior is very difficult for many AI but Dixiter can perform. Dixiter has HINERI engine inside and it enables the behavior.

**Key Words:** component; formatting; style; styling; insert (key words)

## 1. Introduction

Artificial Intelligence: AI has the long history. AI in these days is called as the 3rd generation. Each generations are happened every 10 or 15 years. The 1st generation was in about 1980. Rule-based learning, decision tree 1) and general liner model 2) were researched. During the 2nd generation during about 1990 to 2000, Kernel method 3) and Collaborative filtering 4) were invented. Support vector machine 5), one of the kernel methods, brought big impact to solve pattern recognition problems. Deep Learning 6) is in the 3rd generation. Comparing the 2nd generation, there are two big key differences; big data and computer performance. Deep Learning's basic idea itself has been existing from the 2nd generation as a neural network perspective but this was not succeeded as it's expected. Different from the neural network, Deep Learning involve big data and use much and much computers for its processing. Now on Deep Learning plays main role in this AI generation. Not only computer giant companies but also small startups are researching and developing technologies for Deep Learning.

Beside of such trend, we think that the 3rd generation AI technologies are not still enough for the beyond. Our assumption is that current AI is not enough for Intelligence amplification: IA[1]. This can rephrase that AI does not reach technological singularity[2]. The technological singularity is the point where we come over once, we can come back. This indicates that it has a capability of recursive self-improvement. Once it reaches, the AI autonomously becomes smarter and smarter. It does not require human's interactions.

We explain why current AI is not reachable to the IA by using a simple example. We assume here an AI which can identify a *dog* or a *cat*. For this AI, we must give it data about *which is a cat and which is a dog* to be able to recognize images of them. The data is called as labels. Once the AI has learnt enough, it can identify dog and cat very accurately. But even we have been training continuously, the AI never identifies the difference between *cat* and *lion*. Because we don't define what lion is to the AI. According to this example, IA is that if we give the AI images of cat, dog and lion and teach only cat and lion continuously, it eventually identifies the difference between cat and lion and divided them into different categories.

We think that we can say that current AI technologies can do interpolation[3] but cannot do extrapolation[4]. In general, interpolation is a method to find and put data points from existing data points. And the most important thing is that the newly found points must be placed within existing data points. Extrapolation is the opposite method comparing with interpolation. It finds and puts new data points but they are placed outside of existing data points. This analogy can be applied to current AI. Current AI expects all data should be placed in its closure and focuses how to do accurate interpolations. For accurate interpolation, how to choose borders is very important and it depends on how to teach. Therefore, if we want to identify the difference between cat and lion, we must teach the difference. Extrapolation, here, is that an AI understands that lions should not be placed in its closure and then it extracts a space where it was considered within the closure. After extraction, the space is integrated again as a new space. Figure 1 shows an example of difference between interpolation and extrapolation from AI perspective. I believe that one of required technologies for the beyond is for extrapolation.

In other word, we can say interpolation and extrapolation as learning and discovering. There is an AI developed DeepMind[5] plays Breakout[6] game or Go[7]. The AIs learn many kinds of methods to play the game properly through millions of try-and-error. Some of the methods are not obvious and unexpected and it is very surprising. However the AIs still play inside learning region. They are learning many methods to maximize the effectiveness under the game rules. They never do such discovering like breaking walls in Breakout or placing stones by ignoring the rule of Go during their game.

HIRAMEKI is the key component to realize IA in the beyond. Figure 2 shows the corresponding HIRAMEKI and current AI. This loop is the IA. Current AI consumes a thing and produces common sense (Playing within interpolate-able area). HIRAMEKI consumes common sense and produce non-trivial thing (Get out from interpolate-able area (Extrapolation))

[1] https://en.wikipedia.org/wiki/Intelligence amplification
[2] https://en.wikipedia.org/wiki/Technological singularity
[3] https://en.wikipedia.org/wiki/Interpolation
[4] https://en.wikipedia.org/wiki/Extrapolation
[5] https://www.deepmind.com/
[6] https://en.wikipedia.org/wiki/Breakout (video game)
[7] https://en.wikipedia.org/wiki/Go (game)

However, once, a non-trivial thing is extracted, it becomes a (trivial) thing. Then, the thing can be an input for current AI (Expansion of Interpolate-able area).

However still it is early stage to define what HIRAMEKI is. So in this paper, I'd like to give you a hit of HIRAMEKI. That is HINERI engine. Also an example of HINERI engine adaptation is described as Dixiter; a card game AI of Dixit[8].

The rest of the paper is structured as follows. In Section 2, we introduce data format named the Resource Description Framework (RDF) and its data store which is used by HINERI engine. Section 3 presents the principle idea of HINERI engine and we describe Dixiter adaptions in Section 4. We conclude the paper in Section 5.

## 2. RDF store: Cerise

Cerise, an RDF store that is able to dynamically adapt the storage according to data utilisation patterns. Cerise eciently parallelises query processing onto many servers and optimises disk I/O by co-locating data that are frequently accessed together.

The fundamental data structure of RDF is a Triple. Each RDF triple consists of three elements, namely subject, predicate and object. The subject of a triple represents an RDF resource (identified by a dereferenceable URI). The object can be either an RDF resource or a concrete piece of information (e.g. a String or a Date). The predicate states the relationship between subject and object. A set of triples effectively presents a Graph where subject and object become vertices and predicate a directed and labelled edge connecting the subject and the object. Even though open datasets are fairly static, when these datasets are used as a bridge to link other (potentially private) datasets, the graph can become very dynamic with new edges being created/updated all the time.

### 2.1 Design

Data in the RDF format can be queried using a relational algebra language named SPARQL (SPARQL Protocol and RDF Query Language). A SPARQL query consists of a set of triple patterns where triple elements can be replaced by unbounded variables, e.g. *{s, p, ?o}* (where *?o* is the unbounded variable). Candidate results retrieved with triple patterns are then joined, projected, or filtered. The join semantics of triple patterns can be simplified into *Traverse* queries. Other simple triple patterns need to be expressed as *Scan* queries and then joined with other parts of the query. We design the Cerise system based on this observation. For each type of queries, Cerise provides a dedicated data store and query engine for effcient data processing.

A traverse query is anchored at a specific starting vertex (resource) with a series of constraints to indicate the qualified paths in the graph. Listing 1 presents exemplar triple patterns that can be converted into traverse queries. If either the subject or object of a triple pattern is grounded, that pattern forms part of a traverse query. If all elements of a triple in a query are specified, we can identify a unique triple. For example, the triple fs; p; o:g is the minimal query. By using this query, one can check whether a particular statement exists in the RDF store.

A Scan query is a query where a specified starting vertex is not given. In Listing 2, with {?x p ?z.}, only predicates (edges of the graph) are instantiated and {?x ?y ?z.} specifies the entire triple pattern with unbounded variables.

```
{ s p o . }
{ ?x p o . }
{ ?x ?y o . }
{ s p ?z . }
{ s ?y ?z . }
```

Listing 1: Variation of traversal queries.

```
{ ?x ?y ?z . }
{ ?x ?y ?z . }
```

Listing 2: Variation of scan queries.

### 2.2 Architecture

Cerise consists of three main parts (as shown in Figure 3), the *Query Processor*, the *Graph processor*, and the *Distributed Data Store (DDS)*.

Client applications issue queries through the SPARQL API, which is the query endpoint of the Cerise system. Query Processor relays the SPARQL queries to a SPARQL Query Engine, which parses the query into a set of triple patterns and join operations. The resultant triple patterns and join operations are then translated into Traverse and Scan queries which are issued to the Graph Processor, through the Graph API.

The Graph Processor provides a native Graph Java API. This API, inspired by the Tinkerpop Blueprints API[9], provides a native interface to define graph traverse and scan queries. The Graph Processor also maintains a connection to one of the data servers in the Distributed Data Store. The specified query is then sent to a data server using a socket based proprietary protocol, to the Traverse/Scan API.

The Distributed Data Store is composed by a set of data servers that each include both a Data Store for Traversal (DST) and a Data Store for Scan (DSS). Each one of these components are dedicated to a specific type of query and are distributed among all the data servers. The DSS component is dedicated to execute the Scan queries. The DST component implements the algorithms for the Adaptive Distributed Traverse Query and the Dynamic Data Co-location.

### 2.3 LOD4ALL

Cerise is being built to serve as the backend storage and query engine for LOD4All[10], a public LOD service provided by the Fujitsu Laboratories Ltd. LOD4All caches a plethora of open datasets and facilitate integrated data consumption through one single programming platform. The current version of the service hosts more than 1000 datasets.

## 3. Dixiter

Dixiter plays Dixit. Dixit is A very famous card game in US and Europe. Dixit is a card game by using a deck of cards illustrated with dreamlike images. Players select cards that match a title suggested by the "storyteller", and attempt to guess

[8] https://en.wikipedia.org/wiki/Dixit_(card_game)
[9] http://blueprints.tinkerpop.com/
[10] http://lod4all.net/

which card the "storyteller" selected. The unique rule here is that storyteller must be chosen his/her card by more than one player but not all. Neither no one chosen nor everyone chosen are loss-making patterns for the storyteller. This means that the storyteller must chose “So-so abstracted sentence: SSAS” very carefully and this is the fun part of Dixit.

In Dixiter, HINERI engine contributes to choose SSAS. In the below, we describe detail of HINERI engine and show results.

### 3.1 Principle of HINERI Engine

HINERI Engine tweaks a word to semantically related word. The tweaking process is along graph structure.

There are many data sets which expresses by RDF. One of big datasets is DBPedia, dump data of Wikipedia. In DBPedia, for example, links between page and page are predicates in RDF. In RDF, there are some predicates which indicate similar things like *http://www.w3.org/2000/01/rdf-schema#seeAlso* and *http://www.w3.org/2004/02/skos/core#related*. HINERI engine uses the predicates to tweak a word.

### 3.2 Adaptation to Dixiter

The procedure of Dixiter is following; First, Dixiter extract an exact word from a card. The exact word can show a component, color or behaviour in a card. Second, Dixiter retrieve some URIs which associated with the exact word from LOD4All. Once choose a URI from them, Dixiter query a SPARQL query which starts from the URI and traverse along the predicates which indicate similars. After executing the query, Dixiter can some candidate for tweaked words and choose one from them.

The example query of the SPARQL is in Listing 3. We assume Dixiter extract Tokyo from a card. Then one URI candidate for Tokyo is *http://dbpedia.org/resource/Tokyo*. The query starts from the URI and follow some edges.

```
SELECT ?q
WHERE{
  http://dbpedia.org/resource/Tokyo rdfs:seeAlso ?p .
  ?p rdfs:related ?q .}
}
```

**List 3**: Dixiter sample query

This is a sample of 2 hop tweak. If more tweaking word is needed, we can add more hops. If you want to fit players background, we can specify data set which is used for the query. For example, we can think DBPedia is common sense. It means that the query over DBPedia is for all.

### 3.3 Result of HINERI Engine

We show some results which produced by Dixiter. Can you follow Dixiter?

*Samsung* has been traversed along *Apple→Apple Inc → iPhone → Galaxy → Samsung*. *Pink Floyd* has been traversed along *Sunflower → Sunflower(Album) → The beachboys → Rock Band → Pink Floyd*.

## 4. Conclusion

Dixiter is an AI for Dixit. Dixiter is HINERI engine inside. Current AI engine can produce straightforward word against an image. Straightforward words are not sufficient for Dixit. Bit HINERI engine can tweak to a word to semantically related word and it’s very important for Dixit.

Moreover there is a reason why HINERI engine chooses the word. If you randomly selected, it’s out of blue. We can provide very similar engine by using Word2Vec[11] distance metric but they don’t give us any reasons. If an AI does not provide a reason, the AI reduce funs of Dixit.

## *REFERENCES*


1) Quinlan, J. R. (1987). Simplifying decision trees. Int. J. Man-Mach. Stud., 27(3):221–234.
2) Johnson, R. A. and Wichern, D. W., editors (1988). Applied Multivariate Statistical Analysis. Prentice-Hall, Inc., Upper Saddle River, NJ, USA.
3) Aizerman, M. A., Braverman, E. A., and Rozonoer, L. (1964). Theoretical foundations of the potential function method in pattern recognition learning. In Automation and Remote Control,, number 25, pages 821–837.
4) Sarwar, B., Karypis, G., Konstan, J., and Riedl, J. (2001). Item-based collaborative filtering recommendation algorithms. In Proceedings of the 10th International Conference on World Wide Web, WWW ’01, pages 285–295, New York, NY, USA. ACM.
5) Cortes, C. and Vapnik, V. (1995). Support-vector networks. Mach. Learn., 20(3):273–297.
6) Hinton, G. E., Osindero, S., and Teh, Y.-W. (2006). A fast learning algorithm for deep belief nets. Neural Comput., 18(7):1527–1554.
7) H. Kobashi, N. Carvalho, B. Hu, and T. Saeki. Cerise: an RDF store with adaptive data reallocation. In Proceedings of the 13th Workshop on Adaptive and Reflective Middleware (ARM '14). ACM, New York, NY, USA, pages 1-6.


[11] https://en.wikipedia.org/wiki/Word2vec

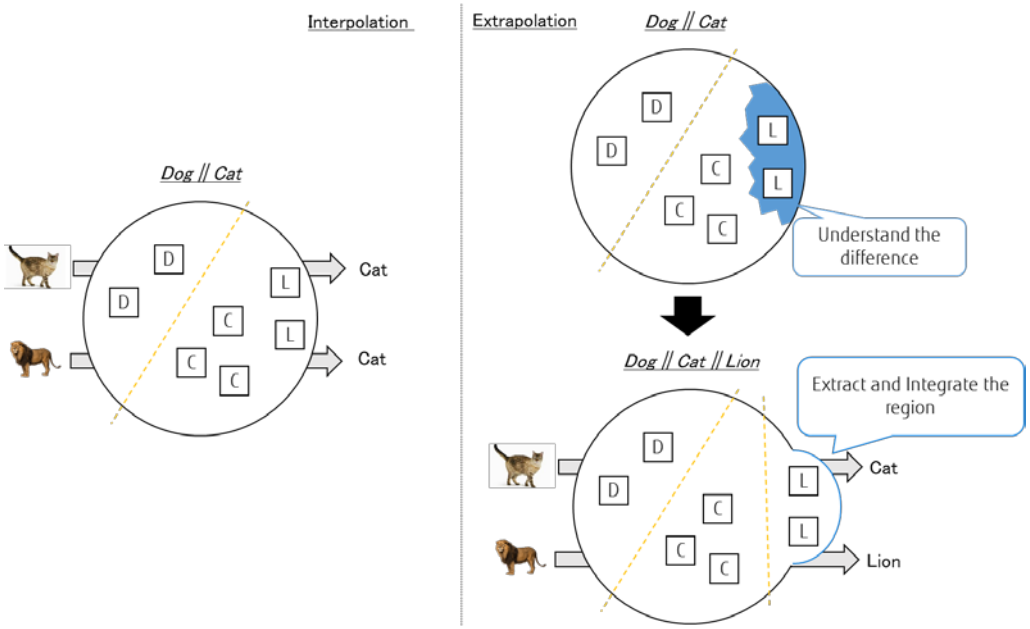


**Figure 1:** Interpolation and Extrapolation. In a sense, we can say the current AI focuses on interpolation because it find out a place in which an unknown thing will be placed among known things. But it can not find out new place out of its closure automatically. Extrapoation is doing that new place finding. In this example, an (interpolatable) AI which know dog and cat, it categorises cat and lion into same place. On the other hand, an extrapolatable AI categorises lions as new type of cat automatically and eventually.

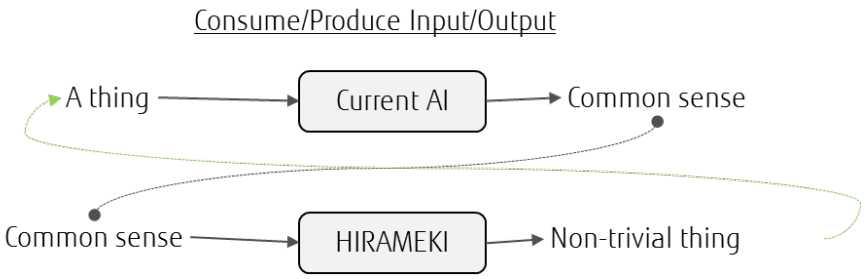


**Figure 2:** HIRMAKEI and AI. This is the IA loop. The IA loop cannot be existed by using only current AI because it produces fix common sense. It does not break through its border. Key points, here, are that 1. A current AI and HIRAMKIE share their output-input, 2. HIRAMEKI produces non-trivial thing but it will be trivial once it's extrcted. These enable the loop.

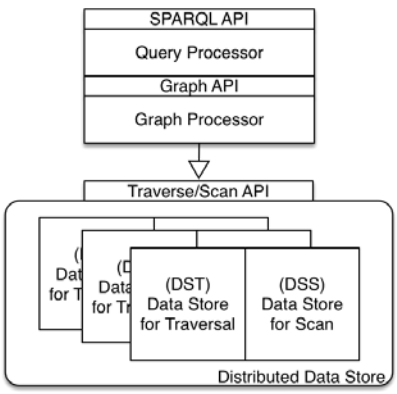


**Figure 3**: System components of Cerise