Takeda Award 2001 Achievement Fact

Techno-Entrepreneurial Achievements for Social/Economic Well-Being

Technical Achievement:

The origination and the advancement of open development models for system software — open architecture, free software and open source software.

The prize is awarded jointly to Ken Sakamura (University of Tokyo), Richard M. Stallman (Free Software Foundation) and Linus Torvalds (Transmeta Corporation).

Ken Sakamura is honored for developing and promoting the TRON open architecture, a realtime operating system specification for embedded systems.

Richard M. Stallman is honored for starting the free software movement and leading the development of the GNU operating system.

Linus Torvalds is honored for developing the Linux operating system kernel by the open source process for software development.

(Awardees are listed in alphabetical order.)

〈Executive Summary〉

The Takeda Award 2001 for social economic well-being is awarded to Ken Sakamura, Richard M. Stallman and Linus Torvalds for the " origination and the advancement of open development models for basic software — open architecture, free software, and open source software. " This achievement has brought the outstanding results through freely/open methods for the development of computer operating system and related software.

In 1980, Ken Sakamura proposed the basic concept of TRON (The Real-time Operating system Nucleus), the operating system kernel for real-time embedded systems. Based on his proposal, the TRON project started in 1984 under the guidance of a committee that was open to anyone. The distinctive features of the TRON concept are that:

- only the application programming interface specifications are defined;
- these specifications support real-time operations and a wide-range of processors,
- and
- the resultant specifications are open to everyone as “ open architecture. ”

In each step during the project cycle -- at the first definition of the specifications, and during refinement of specifications after the manufacturersㄱ implementation of the specifications, the process is open in order to gather the wisdom of many users. Resultant TRON specifications have been widely adopted and implemented into various kinds of embedded systems installed in personal/home electronic products, cellular phones and automobiles.

In 1984, Stallman started to write Unix-compatible system software as free software. This was the start of the GNU project. GNU is a recursive acronym for “ GNU's Not Unix. ” The free software concept includes the freedom to access the source code, to modify the program, to redistribute copies, and to use the modified program. This process has allowed many programmers to contribute improvements to programs. Resultant GNU software, such as a high performance editor, a compiler and a graphical user interface, gained a high reputation for free software because of their excellent functionality and high stability. GNU is the forerunner of the recent open source movement.

In 1991, Torvalds released on the Internet his first version source code of the computer operating system kernel, Linux. The Linux project has been conducted in a unique manner to improve this original version. Many volunteers from around the world have joined the project to improve and debug the opened version of Linux source code. Their improvements were filtered very often by Torvalds to be adopted into the next version of Linux. This process led to to voluntary, enthusiastic and speedy improvements to Linux, and consequently, a high performance and high reliability operating system kernel was produced.

The combination of Linux and GNU software has resulted in a complete set of free/open programs making up the UNIX compatible operating system. Because of their excellent performance and low system development cost, GNU and Linux have begun to be widely used for PC, servers and other general-purpose computers.

These projects,-TRON, GNU and Linux,-used new methods to develop computer system software, which is the key component to information technology. The free/open development styles and the free/open utilization schemes in these projects gathered the wisdom of many people and led to new computer applications and high-grade programs. Furthermore, these projects showed that the free/open development models are very powerful. These models have had an important influence on the computer software market to modify the conventional approach.

Engineering intellect and knowledge emerging from this different scheme from the usual market economy approach is stimulating techno-entrepreneurship in markets and is providing major value to people.

Achievement and Creativity

1. The importance of computer in the information society and the role of operating systems

Recent rapid development of the information technology, especially computer and communication technologies, makes it possible to access the variety of data in the world and, as a result, brings value to people. In the field of the information technology, the software technology is essential as well as the hardware technology. Application software and contents are familiar and useful to users. In addition to stand-alone computers, there are many kinds of intelligent electronic products around us. The number and variety of processor-embedded systems is greater than that of stand-alone computers.

Information moves around the world and it is desirable that it be accessible by people everywhere. It is desirable to have hardware and software for information technology be used by all people who want them. In the information society, " being free/open ", that is, being freely accessible and being freely usable is an especially important feature. " Being free/open " is also important in the development of computer software, especially system software, in order to gather and use the wisdom of many people.

In the computer software system, operating systems are usually used to make it easier to develop application software and contents. Operating systems are the set of common and basic functions. Recently, even in small-size embedded systems, which usually have severely limited memory capacity , it has become popular to use operating systems to improve the performance, reduce the cost and improve the reliability of applications software.

For example, Windows, Mac OS and UNIX are operating systems whereas a word processor and a spreadsheet are application softwares. Operating systems provide fundamental and common functions. Editors and compilers are examples of basic software included as operating systems utilities to help the application software development. Using operating systems, the application software developer can raise the development efficiency and unify the software operation style. Thus, operating systems are the common and basic software systems, which are the key components in the software in the infrastructure of the information society, computers and intelligent electronic products.

Operating system performance is dependent on the ability and personality of the programmer and operating systems differ in their speed and stability. Operating systems have been increasing in size and developing in functionality, along with improvements in hardware and other software. As a result, defects and low performance modules are more likely to become incorporated in operating systems. Countermeasures to avoid these problems become very important and the operating system specifications must be optimized according to the hardware ability and the application field.

The tendency toward the increase in software size and development cost has contributed to the domination of proprietary software. The software production companies have prohibited users from changing programs or accessing their source code. However, any large-scale software cannot avoid the software troubles and bugs, which sometimes seriously hurt computer performance. This kind of influence is most severe for operating systems or the basic software. This has increased the demand for freedom to access and change the source code.

In the case of the small embedded systems such as for home electronic products or cellular phones, conventional operating systems are too big and are less useful because of they lack real-time performance, which realizes high-speed and time-estimative operations for input signals from outside. Vendors distribute some kinds of specific operating systems for that purpose. But, because of differences between the specifications of these operating systems, there are problems. Programmers are easily confused, application software is not as reusable, and, consequently, the software productivity is low. In the case of multi-processor system with various kinds of processors, another problem arises because each operating system is only optimized to the specific type of processors and the unifying the operating systems results in a decrease of total performance.

This achievement, the origination and the advancement of open development models for system software in usual computers or in embedded systems, has resolved these problems. These three projects based on this concept are explained as follows. Table 1 and Figure 1 comparatively show the target field and the characteristics of the three projects. Figure 2 shows the concise development history of each.

2. TRON project

2.1. Summary

In 1980, Ken Sakamura proposed the basic concept of TRON (The Real-time Operating system Nucleus) at the committee of JEIDA (Japan Electronic Industry Development

Association.) After four years of discussion in the committee, the TRON project started in 1984. The basis of the project was the ITRON (Industrial TRON) sub-project, where new operating system interface specifications with superior real-time characteristics for embedded systems was discussed, decided and made available as " oarchipen tecture " for anyone who want them.

The development cycle, which consisted of specification planning, implementation to various kinds of processors, and specification refining, was repeated. In this cycle, the specifications and implementation results were made open in order to gather the wisdom of many people. This process led ITRON specifications to become the standard accepted by many people.

In this standardization process, the balance between unification and optimization ability was stressed. For adaptability to wide-range of processors, from 8 bit to 32 bit, Sakamura carefully conducted the project on the " weak standardization " principle to avoid the over-specification, which degrades the performance in different hardware architecture processors.

In 1988, TRON association was founded in Japan and accelerated the ITRON specification promotion. Microprocessor manufacturers and other companies moved to implement the embedded-system oriented operating systems based on the decided ITRON specification[1, 2] into various kinds of microprocessors.

Using this ITRON specification, standardization in application software programming has been developed for embedded systems. This standardization has been applied in home electronic products, cellular phones, automobiles, facsimiles and digital cameras and other products.

The TRON association North America Liaison Office has been founded in US, and the Korea TRON association was founded in August 2000. They are internationally spreading the TRON specification.

2.2. TRON project progress

The first version of ITRON specification was distributed in 1987. After considering the results and of the implementation of the first version, and views expressed by many people, the next version was divided to two kinds of specifications and both were distributed. One was the uITRON2.0 specification that was restricted in functionality for smaller microprocessors, and the other was the ITRON2 specification that was designed for larger-scale microprocessors.

The μITRON3.0 specification that was made available in 1993 was developed to be uniformly applicable to a wide variety of hardware from 8 bit to 32 bit processors, which further accelerated application standardization. Additionally, that version included new specifications for network-linked distributed systems, which enabled ITRON specification to be responsible to the network development.

Utilizing the results of ITRON, other TRON sub-projects[3] have been progressing; BTRON for operating systems and related specifications for human-machine interface in PCs or workstations; CTRON for operating systems for communication equipment; and HMI for human-machine interface guidelines applicable to all kinds of embedded systems.

Recently, in 1997, JTRON specification has been made available, which combines ITRON with the Java execution environment that is superior in portability and network transparency. In 1999, μITRON4.0 specification has been made available, which raises the program compatibility by the “ standard profile definition. ”

2.3. Results of TRON implementation

ITRON specification operating systems are available for most kinds of microprocessors in the world. ITRON OS has the largest share in the embedded system in Japan[4], according to the results of the questionnaire to Japanese users by TRON association. From 30 % to 40 % of embedded systems use ITRON[5]. Especially, for home electronic products, 80 % of systems in Japan used ITRON in 1999. Recent typical examples of ITRON implementation are the cellular phone by NTT Docomo and the automobile by TOYOTA.

3. GNU project

3.1. Summary

In 1984, Richard M. Stallman resigned MIT and started to develop a UNIX compatible operating system as free software. This was the start of the GNU project, whose object was to make and distribute the free software. In 1985, he made available on the Internet a free editor program, and founded the FSF (Free Software Foundation[6]) as the association for accelerating GNU project.

The GNU project is the forerunner of the freely open method for computer software development. The free software concept proposed by Stallman includes the freedom to access to the source code, to modify the program and to redistribute copies. This freedom and open principle made it possible to gather wisdom from many programmers around the world, resulting in the development of system software with excellent functionality and high stability. GPL (General Public Licensing)[7] is the refined license document summarizing this principle. Based on this principle, Stallman planned to develop UNIX compatible software system. As the result, GNU software, such as a high performance editor (Emacs), a compiler (GCC) and a graphical user interface (GNU desktop: GNOME), has been made. An operating system kernel was under development, but before it was completed, Linux became available. By combining Linux and GNU software, a complete UNIX compatible operating system is available as free software[8].

The concept of free software and the GPL provides the basis for the development of other free software or open software. GNU software developed by GNU project gained a high reputation for free software because of its excellent functionality and high stability.

3.2. Results of GNU project (GNU software and GNU GPL)

The number of GNU software under the umbrella of the GNU project is up to 800. More than 14 GNU sub-projects are registered and progressing now.

The meaning of “ free ” in GNU free software[9] is not that the price is free but that the software has the freedom as follows. To be able to get a source code is the premise of being free.

a) You have the freedom to run the program, for any purpose. (The 0th freedom)
b) You have the freedom to modify the program to suit your needs. (The 1st freedom)
c) You have the freedom to redistribute copies, either gratis or for a fee. (The 2nd freedom)
d) You have the freedom to distribute modified versions of the program. (The 3rd freedom)

The concept of “ copyleft, ” the coined word by Stallman, means that modified versions must also be free at re-distribution without any restrictions. The above-listed conditions are the actual conditions for copyleft[10].

3.3. Reliability of free software

According to the study of standard UNIX utility programs by Miller in 1990 and 1995, the failure rate of the GNU utilities was only 6 % or 9 %[11]. By comparison, the failure rate of utilities on the commercial versions ranged from 15 % to 40 %. The report of the President1s Information Technology Advisory Committee in September 2000[12], recognized the high quality of free software or open source software. The reason for their high quality was explained as follows[13].

1) In the case of the free software, all community members work cooperatively. Users not only report bugs but also propose the revisions and work together to correct bugs, communicating by e-mail, which results in an efficient revision.

2) Because that the source code is open to everyone, programmers have a high moti vation to raise the reliability and to debug the source code thoroughly.

4. Linux project

4.1. Summary

In 1991, Linus Torvalds made the first version of his original Linux and released it on the Internet[14]. Linux was the UNIX compatible computer operating system kernel. After that, many volunteers from around the world joined the Linux project to improve and debug Linux. Revised versions of Linux were also released with no charge to anyone who wants to use it.

The characteristic feature of Linux development is the introduction and promotion of a new development style[15] where many volunteers freely and directly join in improving and debugging the program whenever they want. Torvalds decides which contribution to accept and to be included in the next revision of Linux. This style did not produce competing versions of Linux and kept the uniformity of the operating systems kernel.[16, 17]. He released new versions frequently to distribute the results of debugging and other improvements. This frequent release of new versions made it possible to effectively manage the willing efforts by volunteers.

This open-style computer software development resulted in voluntary, enthusiastic, and speedy improvements, and led to a high performance and high reliability operating system kernel.

4.2. Linux development progress

In September 1991, Linus Torvalds released the first version of Linux0.01. After that, Version 1.00 was released in 1994, which was the completion of the kernel. Version 2.0, released in 1996, made Linux a practical operating system[18].

Torvalds plays the central role in Linux development. The worldwide volunteers develop by themselves a part of source code that they need and Torvalds decides which source code revisions to adopt with the help of filtering process by his supporting staff. Because of this development process, where any volunteers develop the programs they need, high quality programs can be obtained in a short time.

The Linux specification was based on the UNIX standard specification POSIX. That means Linux is a UNIX clone. But the source code of Linux is original and the Linux source code is freely open for anyone.

The revision record of Linux is as follows[19, 20].

1991: Torvalds started to develop Linux. Release of Version 0.01.
1994: Release of Version 1.00.
1996: Release of Version 2.0 for general use and Version 2.1.0 for development use.
1999: Release of Version 2.2.0 for general use and Version 2.3.0 for development use.
2001: Release of Version 2.4.0 for general use.

From Version 0.01 to 1.00, 119 revisions were released in 2 years and 5 months. From Version 1.00 to 2.0, 234 revisions were released in 2 years and 3 months. From 2.0 to 2.1.0, 21 revisions were released in 4 months. From 2.1.0 to 2.2.0, 142 revisions were released in 2 years and 4 months. From 2.2.0 to 2.3.0, 9 revisions were released in 4 months. From 2.3.0 to 2.4.0, 74 revisions were released in 1 year and 8 months. In the mostly frequent revision period in the year from 1994 to 1996, more than 10 revisions were release per month.

4.3. Feature of Linux distribution

The user-friendly package, which contains installation software, network support software and utility programs as well as Linux itself, is prepared in CDROM or on Web site. There are various types of distribution. Some distributions are no charge by volunteer. Some distributions are commercially sold by companies. This distribution system promotes Linux while keeping the unity of Linux to guarantee the compatibility.

In the case of UNIX, the source code was made available through a for-fee license from AT&T. BSD code, developed in University of California in Berkeley, can also be obtained with a license. Using these source codes, each vendor prepared each commercial package after appropriate revision.

In the UNIX standardization process, the specification like POSIX was promoted instead of the unification of the source code itself. The source code of the operating system kernel was dependent on each vendor. As a result, the source code was different in each vendor and the specification was different in details. UNIX operating system cannot be guaranteed to be the compatibility with each other.

When some trouble happens during the running of an application program, the cause of the trouble cannot be analyzed because the vendors did not make the source code available. It is often impossible to determine the reason without the vendor 's cooperation whether the trouble is in the application program or in the operating systems itself.

Compared with the above-mentioned UNIX case, in the Linux case, each distribution contains the open source code and compatibility with the unified Linux kernel is guaranteed.Technical support can be provided easily because of the open source code. The Linux license is at no charge. This no-charge license becomes a great advantage, especially when many machines are used or large amount of embedded systems are produced.

4.4. Worldwide usage of Linux and GNU software

Linux was at first an operating system for stand-alone desktop computers, and afterwards was improved to support multiprocessor systems. Linux is widely spreading to be installed in various kinds of systems; super computers, large-scale systems, Internet terminals, cellular phones, and other embedded systems. The programmer who feels that a modification is necessary modifies the Linux source code. Other people can then use the resulting open source code.

The Linux share is 25 % (1.3 million) of the worldwide server operating system deliveries in 1999 and 7.8 % of the Japanese server operating systems deliveries in 2000. This means Linux is the dominant server operating system. Because of their no-charge for operating systems itself, the market share based on value of sales is not meaningful. In any case Linux has produced a very large market.

GNU software and Linux have begun to be widely used in server and conventional computer systems, because of their excellent functionality and reliability and because of the low systems development cost. Recently, main computer makers have decided to develop the ability to install Linux in all kinds of their machines. In May 2001, IBM, Fujitsu, Hitachi and NEC agreed to develop an operating system for essential business systems such as bank teller systems, using Linux as a base. Besides the merits of Linux -- low cost, system stability, high network ability and system compatibility -- cooperation by these four companies makes it possible to reduce the development cost and to shorten the development period.

5. Impacts

ITRON, the basis of the TRON project, has been applied in home electronic products, cellular phones, automobiles, facsimiles and digital cameras and other products. From 30 % to 40 % of embedded systems use ITRON and, especially, for home electronic products, 80 % of systems in Japan used ITRON in 1999.

In US, the TRON association North America Liaison Office has been founded and 10 dealers deliver ITRON operating systems. In Korea, the Korea TRON association has been founded in August 2000, and it has opened the home page for inquiries from Korean companies and is promoting TRON specification.

Large numbers of computer systems have introduced the combination of GNU software and Linux. Their share is high in the worldwide server operating system deliveries. Recently, main computer makers have decided to adopt Linux. In December 2000, Linux has been adopted into U.S. Army vehicle monitoring system that needs very high reliability.

The open development method also influences other software development. For example, in web server software development, this method was adopted and the web server software Apache was developed. The share of Apache was 30 % in web server software in 1996, and 60 % in 2000. Apache is the most popular web server software at present.

6. Conclusion

As explained above, these projects, TRON, GNU and Linux, introduced new methods, the free/open development styles and the free/open utilization schemes, to develop the computer system software that is the key component to the information society. Ken Sakamura, Richard M. Stallman and Linus Torvalds are the key persons in these projects. These projects widely gathered the wisdom of many people and led to the development of new computer application with the high-grade programs. Furthermore, these projects showed that the new development models were powerful. They have influenced the computer software market to change from conventional development and distribution processes. Engineering intellect and knowledge grown up in the different scheme than usual market economy is stimulating the techno-entrepreneurship in the market side and is realizing to supply the values to people.

7. CURRICULUM VITAE

Ken Sakamura

1979 Ph,D,keio University
1984 started TRON project
present, Professor of University of Tokyo

Stallman, Richard M.

1971 MIT AI laborltory
1984 resigned from MIT and started GNU project
1985 founded Free Software Foundation (FSF)
present, President of FSF

The origination and the advancement of open development models for system software — open architecture, free software and open source software.

Torvalds, Linus

1988 Helsinki University

1991 developed Linux while in Helsinki University

present, working at Transmeta Corporation

REFERENCES

1) K. Sakamura. The TRON Project. *IEEE Micro*, Vol. 7, No. 2 (Apr. 1987), pp. 8-14

2) K. Sakamura. The TRON Project. *Microprocessors and Microsystems*, Vol. 13, No. 8 (Oct. 1989), pp. 493-502.

3) http://www.tron.org/tronproject/tp_view.html

4) http://www.itron.gr.jp/survey00/result-j.html

5) Introduction to the ITRON Project, http://www.itron.gr.jp/panph98/panph98-j.html

6) www.gnu.org/gnu/thegnuproject.html

7) www.gnu.org/japan/manifesto-1993j-plain.html

8) www.Linux-mag.com/1999-07/stallman_01.html

9) www.gnu.org/philosophy/free-sw.ja.html

10) www.gnu.org/philosophy/categories.ja.html

11) ftp://grilled.cs.wisc.edu/technical_papers/fuzz-revisited.pdf " Fuzz Revisited: A Re-examination of the Reliability of Unix Utilities and Services " by Barton P. Miller et al.

12) http://www.itrd.gov/ac/letters/pitac_ltr_sep11.html " Report to the President on Open Source Software for High End Computing ", September 11, 2000

13) www.gnu.org/software/reliability.ja.html

14) What is Linux?, Linux Journal Home Page, June 24,1999. www2.linuxjournal.com/about_linux/index.html

15) T. Ootani, Linux was born like this, " *Newest Linux Guide with Ease* " Gijutsu Hyoronsha, May 1999, pp102-105.

16) Linus Torvalds and David Diamond, " Just for Fun: The Story of An Accidental Revolutionary by Linus Torvalds and David Diamond ",Shougakukan Production, May 2001.

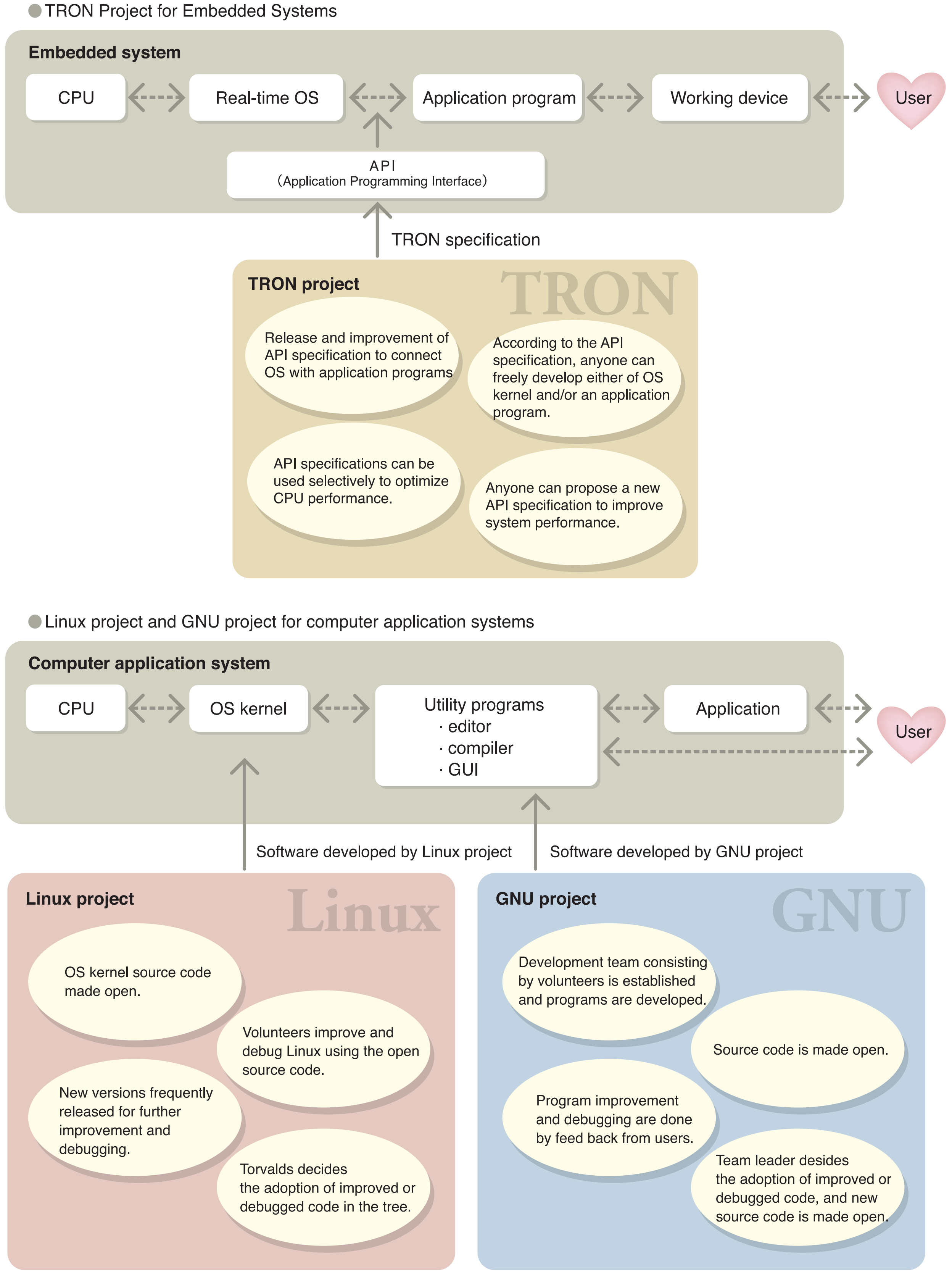
17) Wonderful World of Linux 2.4, Jul 17, 2000, Linuxtoday.com

18) Linux : www.Linux.or.jp/general/Linux.html

19) Linux Kernel Version History, www.memalpha.cx/Linux/Kernel/

20) T. Kishi, " *Linux Kernel History* ", Interface June 2001, p.56. www.cqpub.co.jp/interface/toku/200106/if06_toku1-h1.htm

Figure 1. Comparison of each Project


TRON Project for Embedded Systems
Embedded system
CPU
Real-time OS
Application program
Working device
User
API
(Application Programming Interface)
TRON specification
TRON project
TRON
Release and improvement of API specification to connect OS with application programs
According to the API specification, anyone can freely develop either of OS kernel and/or an application program.
API specifications can be used selectively to optimize CPU performance.
Anyone can propose a new API specification to improve system performance.
Linux project and GNU project for computer application systems
Computer application system
CPU
OS kernel
Utility programs
· editor
· compiler
· GUI
Application
User
Software developed by Linux project
Software developed by GNU project
Linux project
Linux
OS kernel source code made open.
Volunteers improve and debug Linux using the open source code.
New versions frequently released for further improvement and debugging.
Torvalds decides the adoption of improved or debugged code in the tree.
GNU project
GNU
Development team consisting by volunteers is established and programs are developed.
Source code is made open.
Program improvement and debugging are done by feed back from users.
Team leader desides the adoption of improved or debugged code, and new source code is made open.

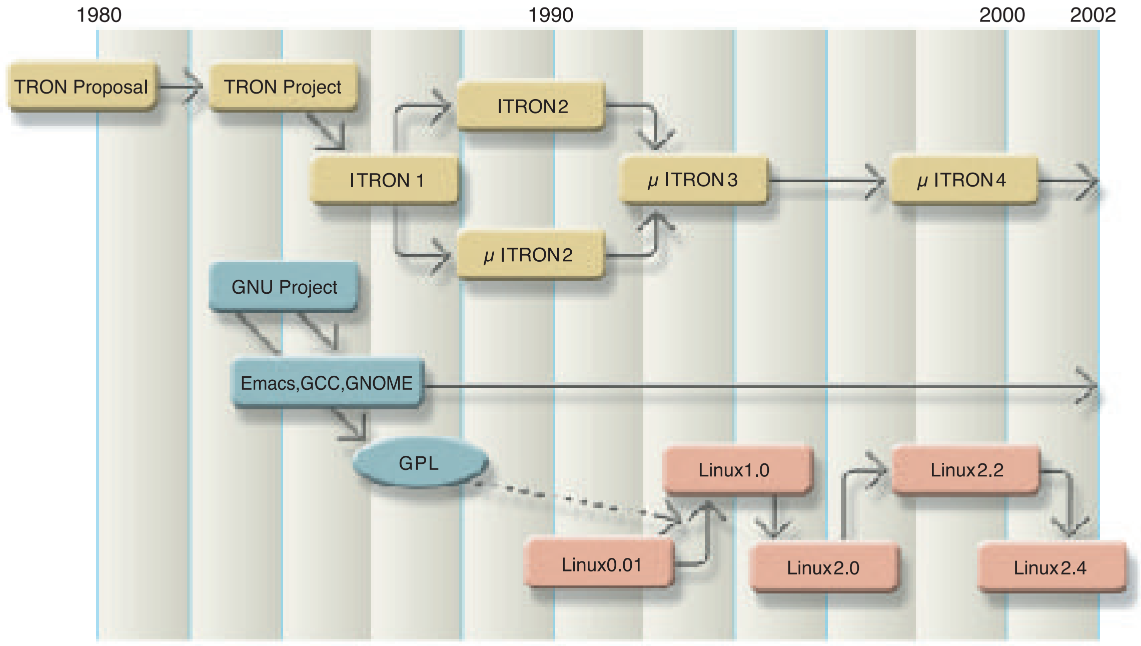
Technical Achievement

The origination and the advancement of open development models for system software
– open architecture, free software and open source software

Table 1. The feature of the development model

	TRON **TRON Project**	GNU **GNU Project**	Linux **Linux Project**
Technical achievement	The OS development model for embedded systems where the API(Application Programming Interface) specification is made open to promote the development of OS and application programs	Development model of utility programs for computer software system and definition of GPL (General Public License)	Development model of POSIX compatible operating system kernel by open source process.
Unique feature of each development model	Open architecture principle of the API specifications. OS kernel and application program can be developed, freely using theAPI specification. Any programmer can propose a new API specification. TRON project decides to include the new API specifications or not.	Project to develop free software UNIX compatible system. Development team consisting of volunteers is established and source code is made open to the public. If modification is made, source code of this modification is required to be made open. Team leader decides if this modification should be included in the new version or not. Anyone can use this program.	Source code of operating system kernel is released. Many volunteers freely work to improve and debug the program.Torvalds decided whether or not to adopt the proposed change. New versions are frequently released to reflect those changes. Everyone can use Linux.

Figure 2. The development progress of TRON, GNU and Linux


1980
1990
2000
2002
TRON Proposal
TRON Project
ITRON2
ITRON 1
μ ITRON3
μ ITRON4
μ ITRON2
GNU Project
Emacs,GCC,GNOME
GPL
Linux1.0
Linux2.2
Linux0.01
Linux2.0
Linux2.4