

ITRON Internatinal Meeting 2000



Overview and Recent Activities of the ITRON Project

27th Sep. 2000

Hiroaki Takada

Toyohashi Univ. of Technology

ITRON Committee, TRON Association

Kiichiro Tamaru

TOSHIBA Corporation

ITRON Project Home Page
<http://www.itron.gr.jp/>

ITRON Project



- ▶ one of the subprojects of the TRON Project
- ▶ a project to standardize RTOS and related specifications for embedded systems
(*esp. small-scale embedded systems*)
- ▶ a joint project of industry and academia
(*not a government project*)

core members:

Fujitsu, Hitachi, Matsushita (Panasonic),
Mitsubishi Electric, NEC, Oki Electric, Toshiba

US companies (or its subsidiaries):

Accelarated Technology Inc. (ATI), Hewlett-Packard,
Metrowerks, Red Hat, U S Software

academia:

Univ. of Tokyo, Toyohashi Univ. of Technology

- ▶ open standard policy

Requirements on Standard RTOS Specification



- ▶ improving software productivity
 - ➔ *easy training of software engineers*
 - ➔ *facilitating the reuse of software components*
- ▶ applicable to various scales and types of processors
 - scalability* – 8-bit to 64-bit MCUs/MPUs
- ▶ deriving maximum performance from hardware
 - ➔ *reducing the cost of final products*
- ▶ truly open standard



The ITRON Specifications have been designed to meet these requirements.

Design Concept and Principles



Design Concept

▶ *loose standardization*

*maximum performance cannot be obtained
with strict standardization*

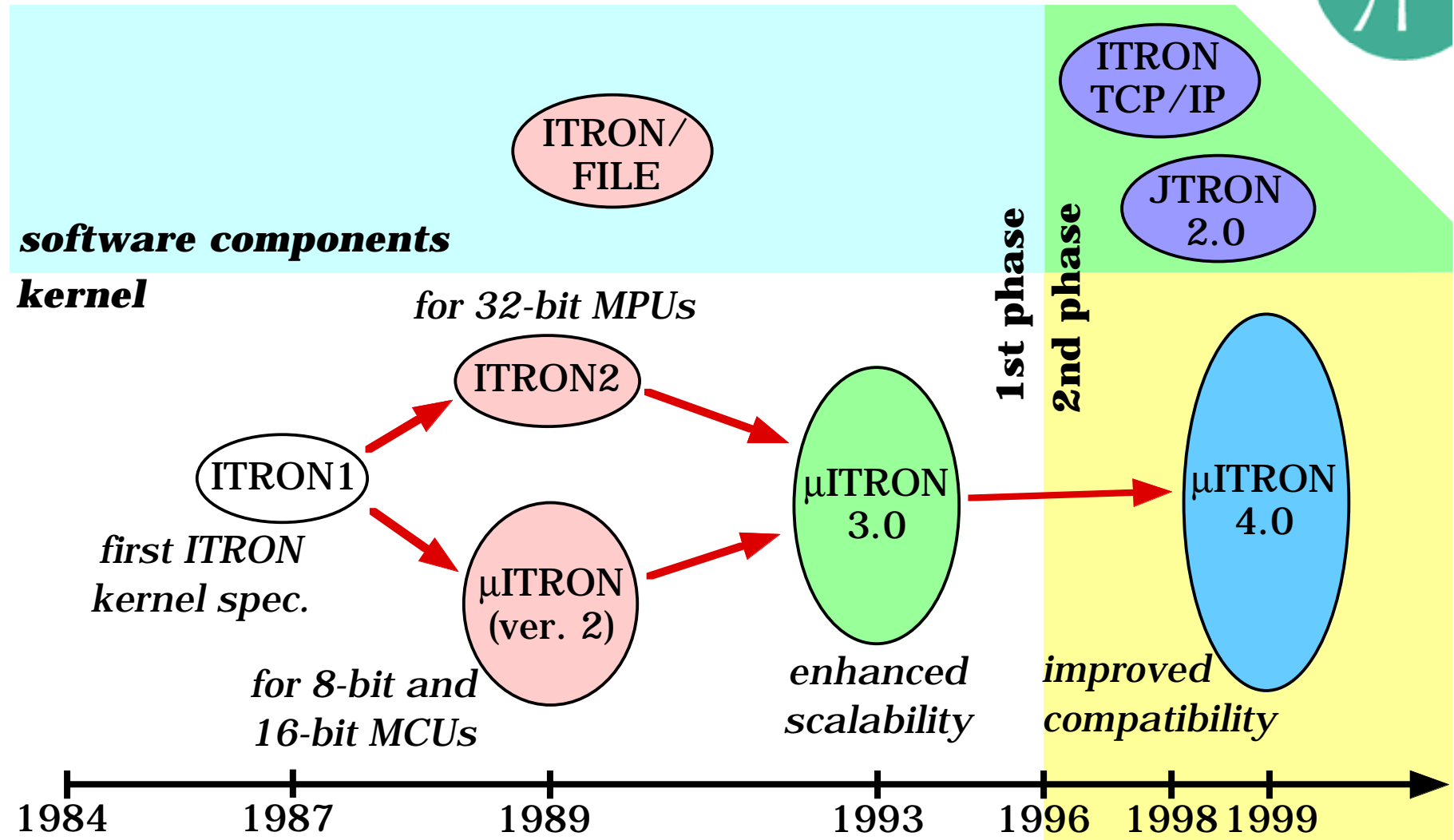


adaptability & scalability

Design Principles

- ▶ allow for adaptation to hardware, avoiding excessive hardware virtualization
- ▶ allow for adaptation to the application
- ▶ **emphasize software engineer training ease**
- ▶ organize specification series and divide into levels
- ▶ provide a wealth of functions

History of the ITRON Specifications



Functions of μ ITRON4.0 Specification



- ▶ task management
- ▶ task-dependent synchronization
- ▶ task exception management
- ▶ basic synchronization and communication
- ▶ extended synchronization and communication
- ▶ memory pool management
- ▶ time management
- ▶ system state management
- ▶ interrupt management
- ▶ service call management
- ▶ system configuration management
- ! *no I/O handling functions defined*

Number of Service Calls

- ▶ full set
 - service calls: 166
 - static API: 21
- ▶ standard profile
 - service calls: 170
 - static API: 11
- ▶ automotive control profile
 - service calls: 43
 - static API: 8
- ▶ minimum set

Implementation Status



- ! *We do not know how many kernels are implemented based on the ITRON specifications.*
- ▶ more than 50 registered implementations for about 40 processors
- ▶ several non-registered commercial implementations
 - ➔ *ITRON-spec. kernels have been implemented for almost all major processors for embedded systems.*
 - 8-bit to 64-bit MCUs/MPUs*
 - ➔ *Some of them are developed by U.S. companies.*
 - U S Software, Red Hat, ATI, and ...*
- ▶ uncountable in-house implementations
- ▶ some freely distributed implementations

Application Status



- ▶ most widely used RTOS specification in Japan
de-facto industry standard
- ▶ widely used especially in consumer applications

Typical ITRON-specification Kernel Applications

Audio/Visual Equipment, Home Appliance

TVs, VCRs, digital cameras, settop boxes, audio components

Personal Information Appliance, Entertainment/Education

PDA's, car navigation systems, electronic musical instruments

PC Peripheral, Office Equipment

printers, scanners, disk drives, CD-ROM drives, copiers, FAX

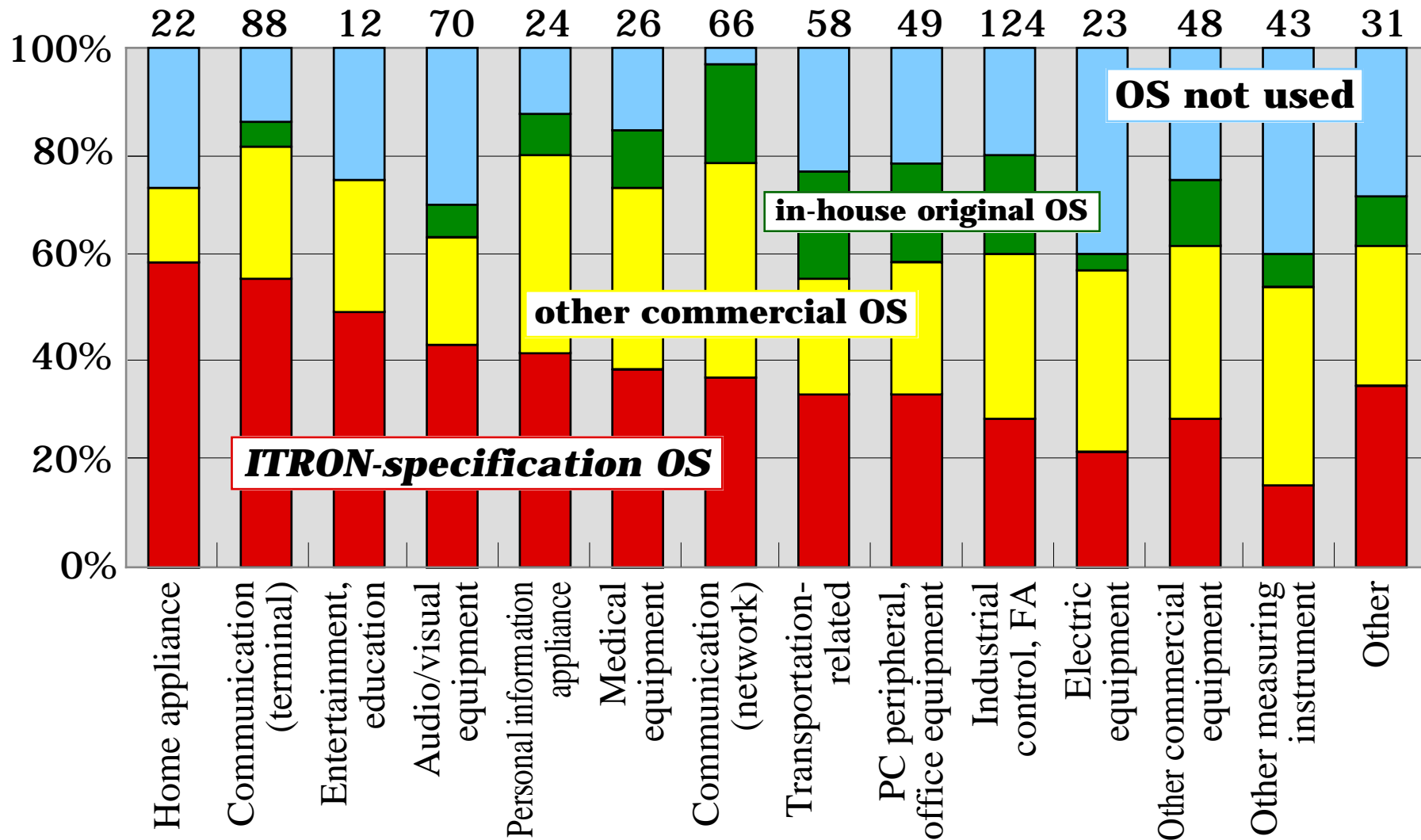
Communication Equipment

cellular phones, ISDN telephones, ATM switches, satellites

Transportation, Industrial Control, and Others

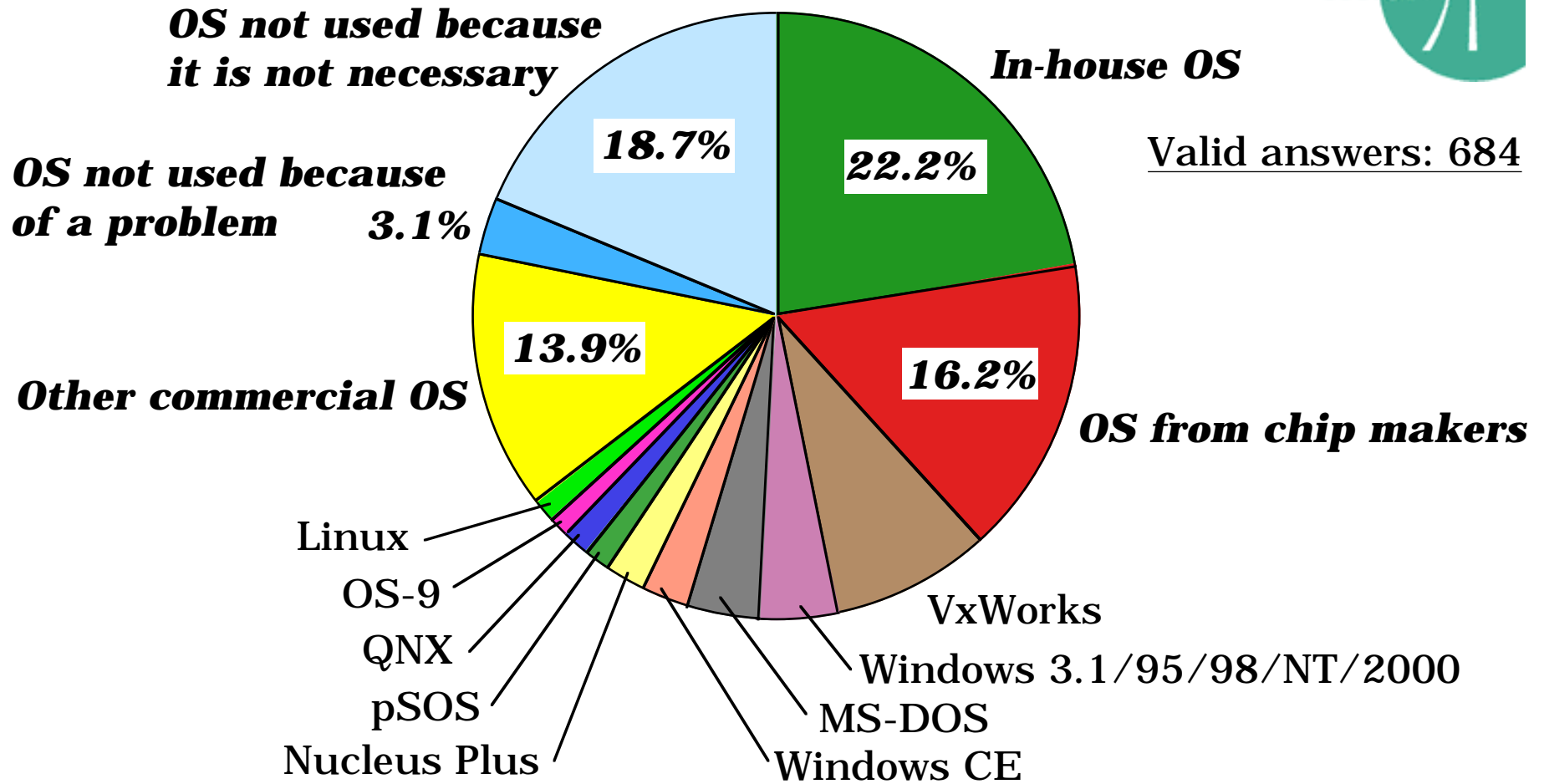
automobiles, plant control, industrial robots, medical equipment

Overview and Recent Activities of the ITRON Project



Embedded OS for each application field
 (TRON Association Survey, late 1999 - early 2000, Japan)

Overview and Recent Activities of the ITRON Project

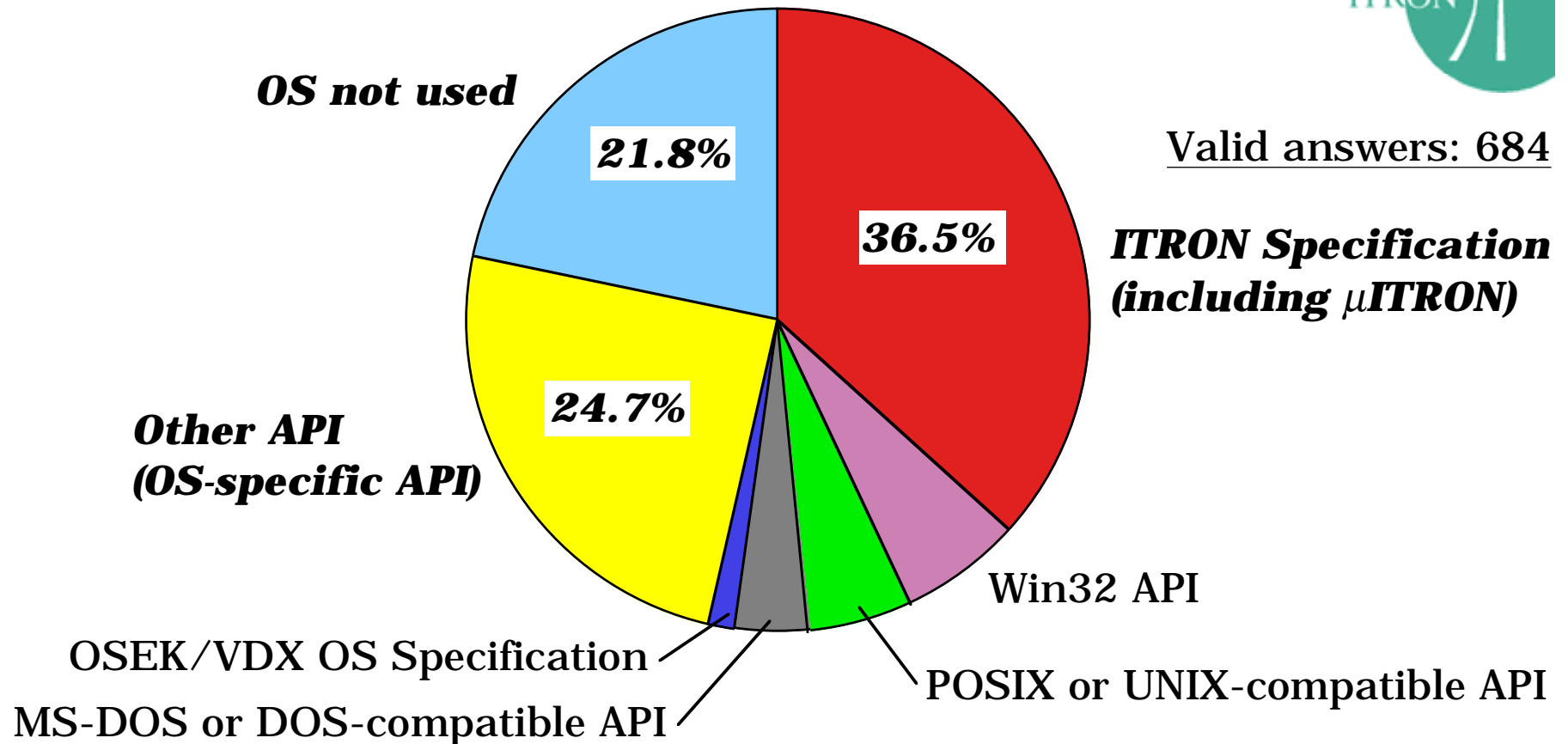


OS Embedded in Recently Developed Embedded Systems
(TRON Association Survey, late 1999 - early 2000, Japan)

Overview and Recent Activities of the ITRON Project



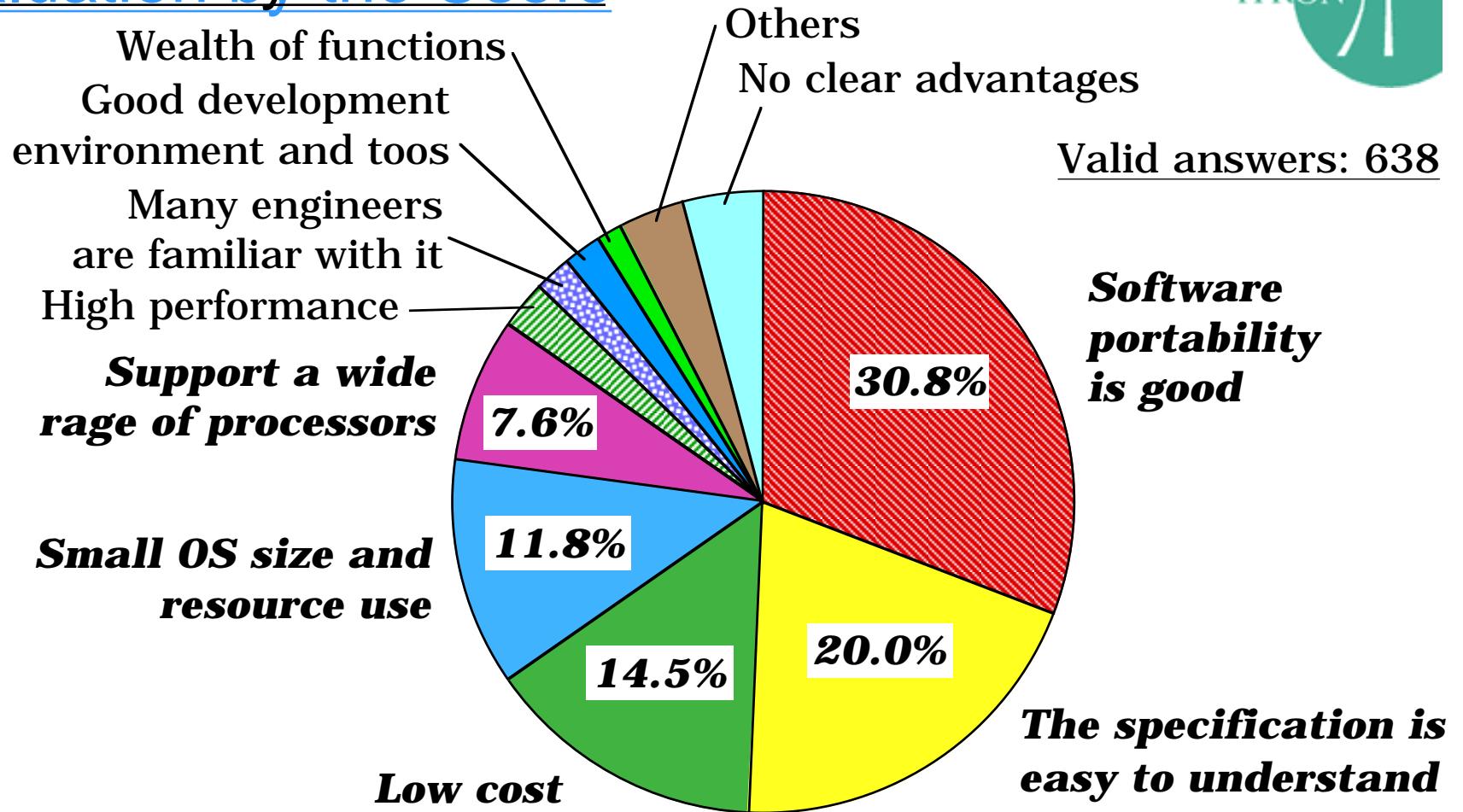
Valid answers: 684



OS API Mainly Used
for Recently Developed Embedded Systems
(TRON Association Survey, late 1999 - early 2000, Japan)

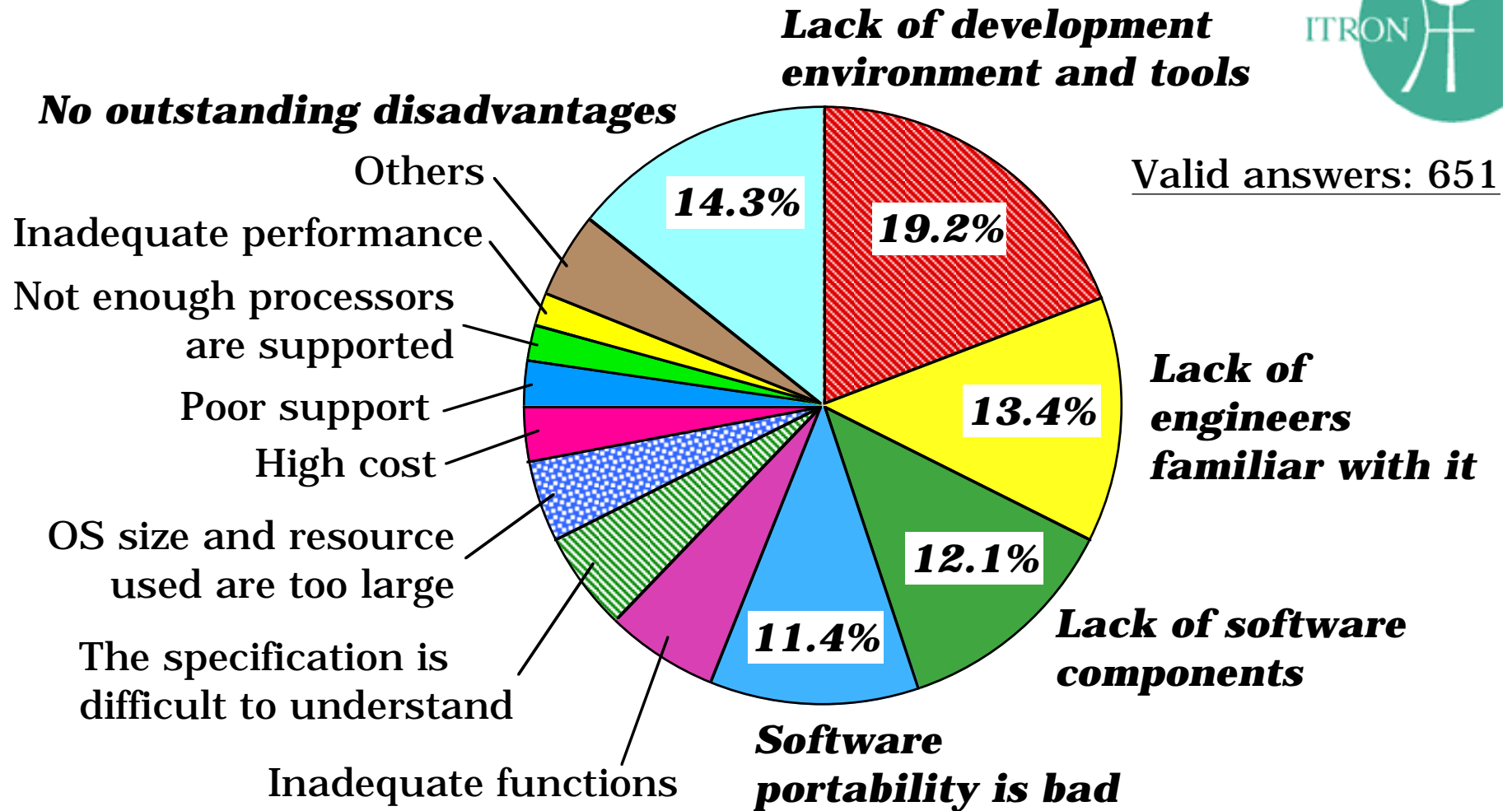


Evaluation by the Users



ITRON-Specification OS Advantages

(TRON Association Survey, late 1999 - early 2000, Japan)



ITRON-Specification OS Disadvantages

(TRON Association Survey, late 1999 - early 2000, Japan)

ITRON Project – 2nd Phase (1996–)



- ▶ broaden the scope of the standardization effort to related aspects

← *Embedded software is getting larger and more complex.*

Software Components (Software IP)

- ▶ satisfying the preconditions for the development, circulation, and use of software components
- ▶ standard API for software components

Development Environments

- ▶ standard interface between kernel and debug tools
- ▶ language binding for C++/EC++

Application-specific Standards

- ▶ satisfying application-specific requirements

Status of 2nd Phase Activities



Preconditions for Software Components

- ▶ μ ITRON4.0 Specification *released in June. 1999*
- ▶ Conformance Testing Method *under investigation*
- ▶ Application Design Guidelines

Standard API for Software Components

- ▶ ITRON TCP/IP API Specification *released in May 1998*
- ▶ JTRON2.0 Specification *released in Oct. 1998*
- ▶ Device Driver Design Guidelines *under investigation*

Development Environments

- ▶ ITRON Debugging Interface Spec. *tentative version released*
- ▶ C++/EC++ Language Binding *under investigation*

Application-specific Standards

- ▶ Automotive Control Applications *reflected to μ ITRON4.0*

ITRON Debugging Interface Specification



Background

- ▶ RTOS makers and software development tool makers are separated in case of ITRON.
 - ▶ *many RTOS makers developing ITRON-specification OSs*
 - ▶ *several software development tool makers*
- ▶ It requires much work to make a debugging tool support many existing ITRON-specification OSs.
 - ▶ *for lack of the standard interface between ITRON-specification OSs and debugging tools*
- weak debugging tool support for ITRON-spec. OSs
 - ↓
 - ▶ **debugging interface standardization**
 - ▶ *standardization effort started in Feb. 1999*



Goal

- ▶ *ITRON Debugging Interface Specification* is to define an interface for adding RTOS-support debugging functions to existing debugging tools.

- ▶ The existence of the debugging environment without RTOS-support functions is assumed.

- ▶ RTOS-support debugging functions:

- ▶ obtaining status of the RTOS objects (tasks, etc.)
 - ▶ obtaining task contexts
 - ▶ invoking a service call from the debugging tool
 - ▶ task-aware breakpoints and stepping
 - ▶ logging/obtaining the RTOS behavior tracks



- ▶ The internal structure of RTOS should be known to support these functions.

Requirements



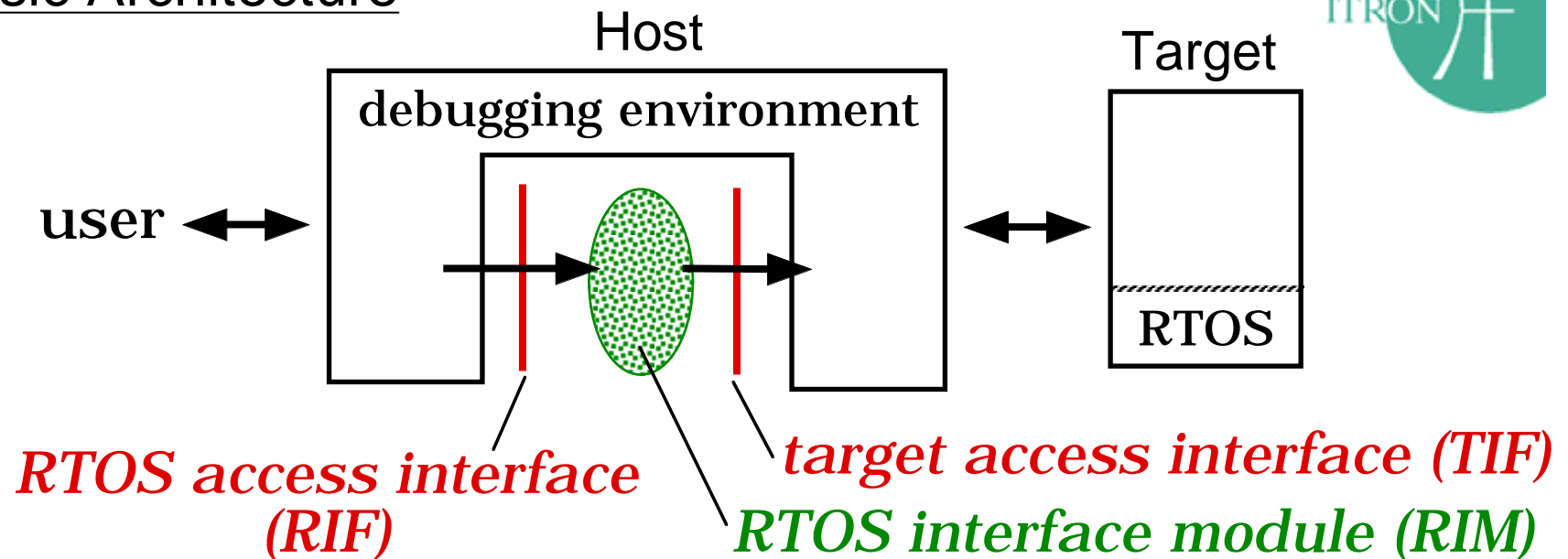
- ▶ **scalability**
 - ▶ The standard interface should be applicable not only 32-bit high performance processors, but also 8-bit and 16-bit low performance processors.
- ▶ **applicable to various debugging tools**
 - ▶ The interface should be applicable to various kinds of debugging tools, including software debuggers, in-circuit emulators, ROM emulators, and so on.
- ▶ **applicable to other RTOSs and software components**
 - ▶ The interface specification should be applicable to other RTOS with small modifications.
 - ▶ The same framework should be applicable to software components.

Possible Approaches



- ▶ standardizing the internal structure of RTOS
 - ▶ **unacceptable** because the RTOS internal structure should be optimized for the environment
 - ▶ adding debugging support functions to RTOS or running debugging task on the target
 - ▶ **unacceptable** because the overhead is large for small-scale systems (eg. systems with 8-bit processors)
 - ▶ standardizing the description method of the internal structure of RTOS
 - ▶ **partial solution** because it is not flexible
- ↓
- ▶ plugging the module which knows the internal structure of RTOS into the debugging environment.

Basic Architecture



- ▶ The **RIM** is the software module supplied by the RTOS maker and plugged into the debugging environment.
- ▶ The **RIF** and the **TIF** are standardized interfaces between the **RIM** and the debugging environment.



Example (1)

- ▶ showing the status of a task
 1. The user requests to show the status of a task.
 2. The debugging environment requests the **RIM** to read the status of the task through the **RIF**.
 3. The **RIM** reads the symbol table and obtains the address on which the task status is stored.
 4. The **RIM** requests the debugging environment to read the address through the **TIF**.
 5. The debugging environment reads the address and returns its contents to the **RIM**.
 6. The **RIM** converts it into the standardized representation of the task status and returns it to the debugging environment.
 7. The debugging environment displays the task status to the user.



Example (2)

- ▶ showing the status of a task when the RTOS supports the service call to read task status
 - 1-2. *<same with the previous example>*
 3. The **RIM** expands the routine to invoke the service call and to write the result to a memory using the **TIF**.
 4. The **RIM** requests the debugging environment to execute the routine through the **TIF**.
 5. The RTOS reads the task status and writes it to the memory.
 6. The **RIM** reads the memory (to which the task status is written) through the **TIF** and returns it to the debugging environment.
 7. The debugging environment displays the task status to the user.
- ***scalability is achieved with RIM***

Current Status



- ▶ The tentative version of the specification was released in July 2000.
- ▶ English version will come soon.
- ▶ Proposing the OSEK/VDX project to jointly standardize the debugging interface of both OSs.
 - ▶ OSEK and ITRON have a similar situation.
- ➔ explain our specification at the 1st OSEK/VDX debug WG meeting on Oct 2nd.

Other Recent Activities



- ▶ Device Driver Design Guidelines
- ▶ Revision of JTRON2.0 Specification to JTRON2.1
- ▶ C++ (*incl.* EC++) language binding
- ▶ μ ITRON4.0 conformance test specification
- ▶ Revision of TCP/IP API Specification coordinative with μ ITRON4.0 Specification (*not started*)

JCG Project



- ▶ Three software components for μ ITRON-specification kernels have been developed
 - ▶ reference implementation of JTRON2.0 specification
 - ▶ CORBA module for small-scale embedded systems
 - ▶ GUI module for small-scale embedded systems




Developped software is distributed in free


! JCG Project is funded by the Japanese government through IPA (Information-technology Promotion Agency)

Next Plan (not technical)



- ▶ North America Chapter
 - ▶ Preparation committee is established in this March.
chair: Donald Dunstan (US Software)
- ▶ TRON Association Restructuring Plan
 - ▶ Name and role of the committees will change
eg) ITRON Committee  Kernel Committee
not fixed

Summary

- ▶ the necessity of open software development in embedded system arena
- 
- ▶ ITRON specification kernel serves as the basis for open development.