ITRON Project Overview

Haruyasu Ito
TRON Association

ITRON Project

- One of the subprojects of the TRON Project
- A project to standardize RTOS and related spec. for embedded systems (esp. small-scale embedded systems)
- A joint project of industry and academia (not a government project)

Core members:

Fujitsu, Hitachi, Mitsubishi Electric, NEC, Oki Electric, Toshiba

US companies (or its subsidiaries):

Accelerated Technology Inc., Hewlett-Packard, Metrowerks, Rational Software, RedHat, US Software

Academia

University of Tokyo, Toyohashi University of Technology

Advantages of ITRON Specifications

- Compact and low-overhead real time kernel specifications
  - fit in a single chip MCU
- Easy to understand
- Open specification
  - anyone can use the specification without any licensing fee
  - complete specification documents on the website (www.itron.gr.jp)
- Applicable to wide variety of processors
  - from low-cost 8 bit MCU to high performance 64 bit RISC
- Widely used for various embedded systems
  - used in over 30% of embedded systems in Japan
- Supported by many vendors

Implementation Status

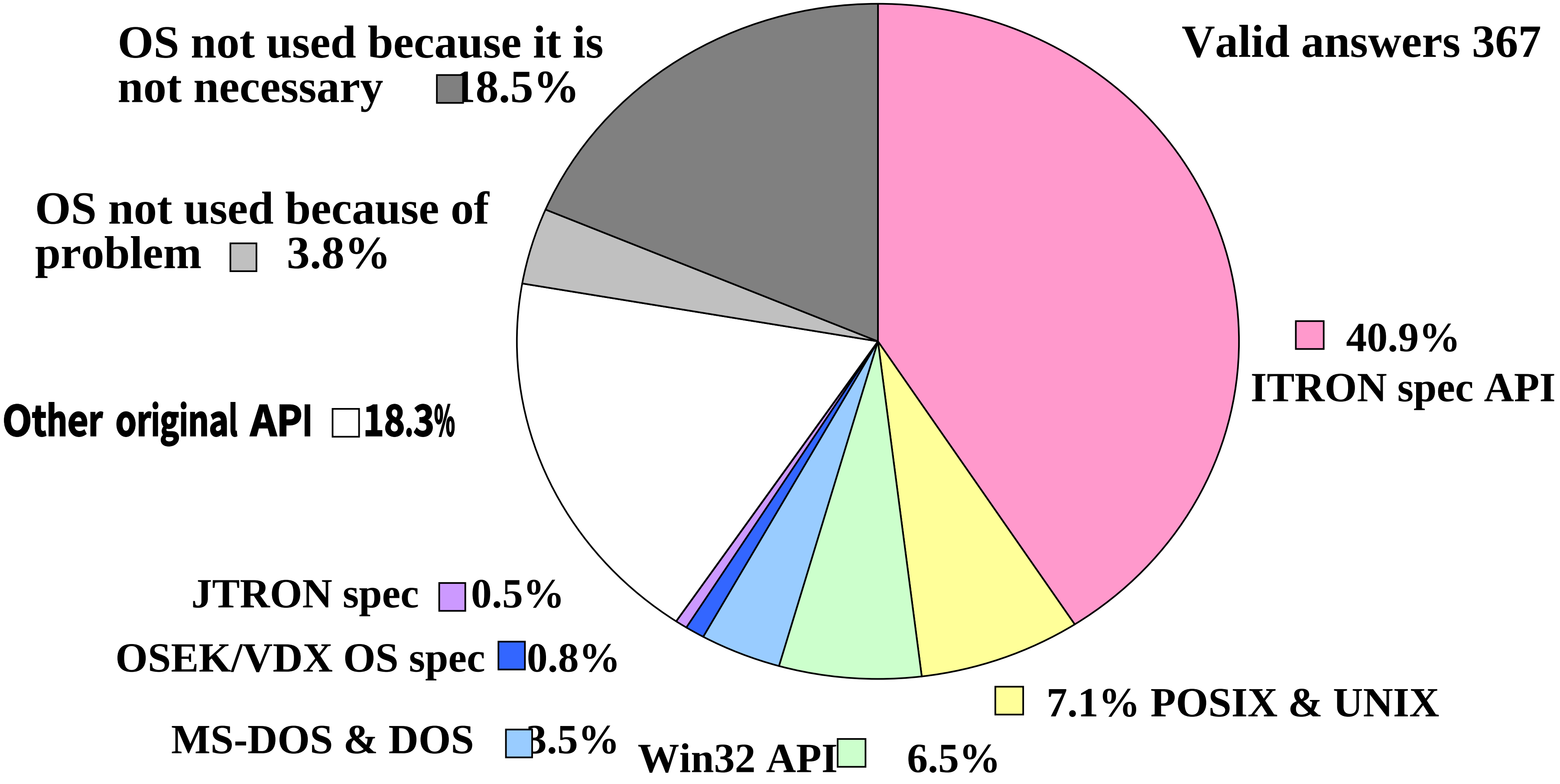
- More than 50 registered implementations for about 40 processors.
- Several non-registered commercial implementations

  => ITRON-spec. kernels have been implemented for almost all major processors for embedded systems. (8-64 bit MCUs/MPUs)

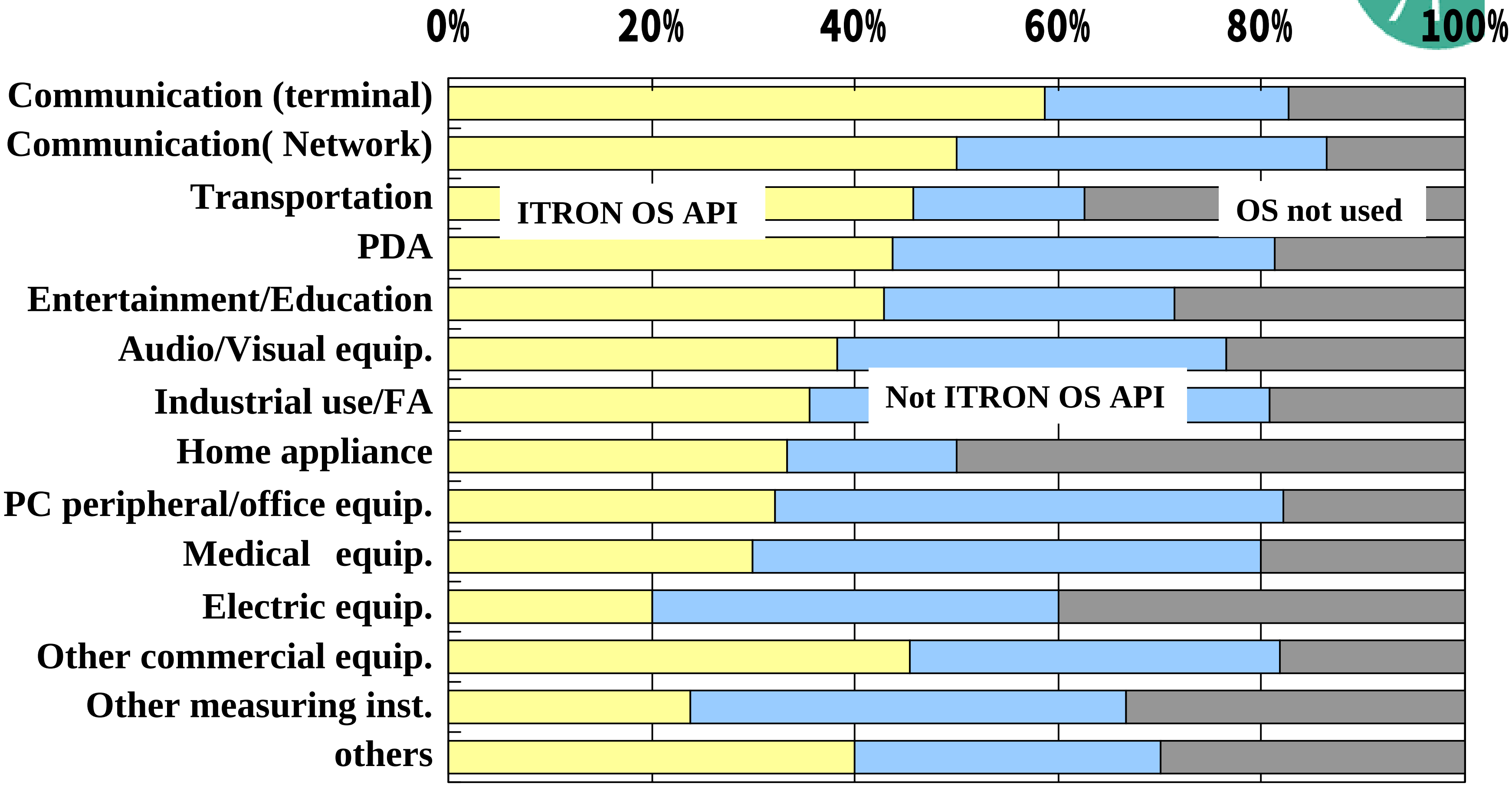
  => Some of them are developed by U.S. companies.

  US Software, RedHat, ATI
- Uncountable in-house implementations
- Some freely distributed implementations

ITRON SPEC. API Share in Embedded Systems (Japan)



ITRON API Use in Embedded Systems in Japan



Valid answers 355

ITRON Application (Engine control)

ITRON Application (Cellular phone)

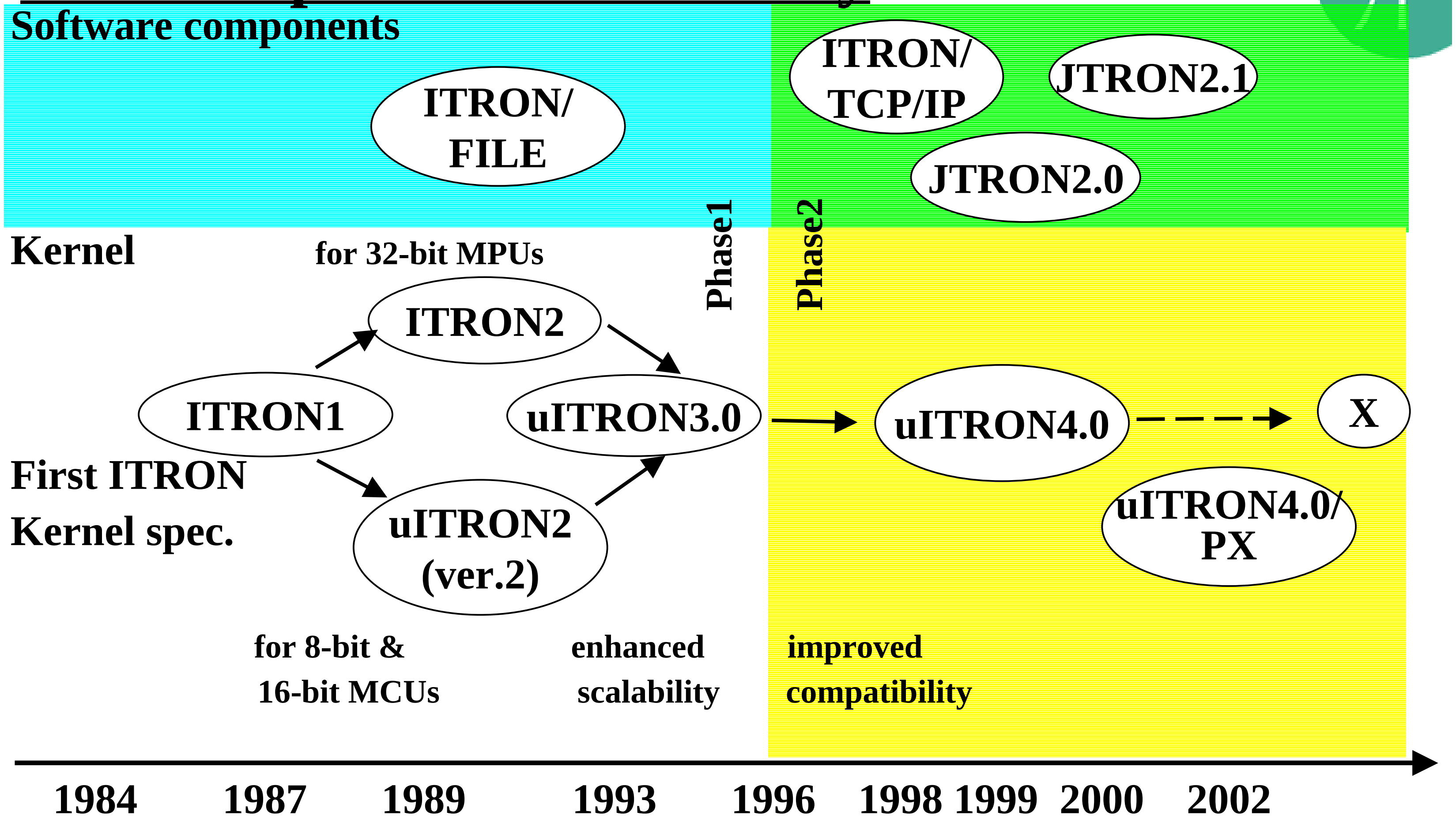
ITRON Application (VCR)

ITRON Application(FAX)

Current Status of ITRON Specification

- Software components
  uITRON4.0 specifications
  Conformance test specification
  ITRON TCP/IP API specification
  JTRON2.1 specification
  Device driver design guidelines (Under investigation)
- Development Environments
  ITRON debugging interface specification
  C++/EC++ language binding (Under investigation)
- Application-specific standards
  Automotive control applications (reflected to uITRON4.0)

ITRON Specifications History



Software components

ITRON/ FILE

ITRON/ TCP/IP

JTRON2.1

JTRON2.0

Kernel

for 32-bit MPUs

Phase1

Phase2

ITRON2

ITRON1

uITRON3.0

uITRON4.0

X

First ITRON Kernel spec.

uITRON2 (ver.2)

uITRON4.0/ PX

for 8-bit & 16-bit MCUs

enhanced scalability

improved compatibility

1984 1987 1989 1993 1996 1998 1999 2000 2002

µITRON 4.0 - What and Why

µITRON 4.0 is the next generation µITRON real time kernel specification

Why it is necessary?

- Software portability
  - Our “loose standardization” policy often contradicts with software portability”
- Functions for independently-developed software components

  Incorporating the results of recent investigations

  - Hard real time systems supports
  - Requirements for automotive control application
- Following the advancement of microprocessor technology

Portability vs. Adaptability

- Portability of software components built on μITRON can be raised if we define the kernel functions more strictly
- Adaptability (incl. scalability) is the most important advantage of μITRON, so it should be kept

↓

Standard Profile

- The set of kernel functions strictly defines for raising software portability

  μITRON 4.0 - loose standardization
  standard profile - strict standardization

- ***Subsetting*** is still acceptable for small systems
- ***Extended functions*** are also defined

Standard Profile - Overview

Target System

- Target processor: high-end 16 bit and 32 bit
- Kernel size: 10kb to 20kb with all functions
- The whole software is linked to one module
- Kernel objects are statically defined

Function Overview (See http://www.itron.gr.jp)

- Includes almost all level S functions of µITRON 3.0
- Incorporates some level E functions of µITRON 3.0
- Includes newly introduced functions
- Several µITRON 3.0 function have been modified; others more strictly defined

Standard Profile - Function Overview (cont)

Level S of μITRON 3.0

- Basic task management and synchronization
- Semaphore, eventflag, mailbox
- Interrupt management, basic time management

From Level E of μITRON 3.0

- Fixed-sized memory pool, cyclic handlers
- Service calls with timeout

Major Modifications / More Strict Definitions

- act_tsk with queuing instead of sta_tsk
- Some terminology and service call names
- How to write an interrupt handler in C
- Service calls used in an interrupt handler

Standard Profile - Function Overview (cont)

Newly Introduced Functions

- Data queue (queue one word messages)
- Exception handling mechanism
  - task exception routine, CPU exception handler
- System state reference
- can_act, isig_tim

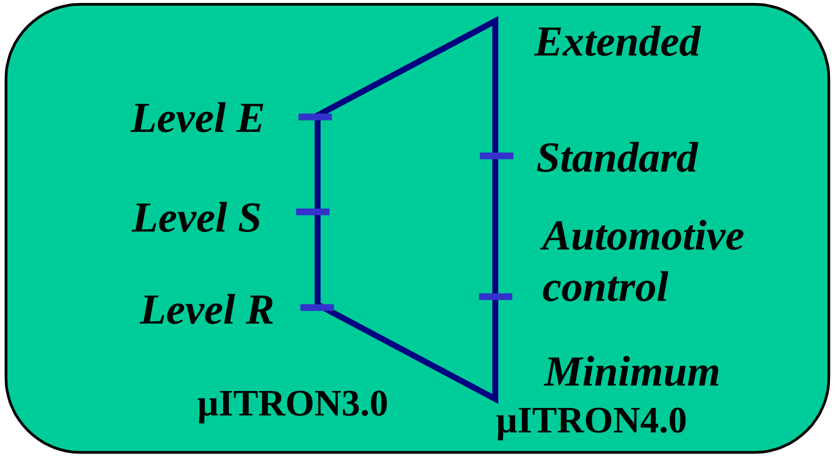
Static API

- Standard description (in a system configuration file) for defining kernel objects statically
  - cre_tsk(…) - service call for creating a task
  - CRE_TSK(…)- static API for creating a task
  - Both of these have common parameters

Broader Scalability

New Functions not Included in μITRON 3.0

- Data queues
- Task exception handling
- System state reference
- Interrupt service routine
- Hard real-time support
- Automatic ID assignment



Automotive Control Profile

- Smaller profile definition especially suitable for automotive control application

Minimum Requirements

- Dormant state instead of waiting state is mandatory

Functions Supported in μITRON 4.0 Spec

- Task management
- Task-dependent synchronization
- Task exception management
- Basic synchronization and communication
  - (Semaphore, eventflag, data queue, mailbox
- Extended synchronization and communication
  - (mutex, message buffer, rendezvous)
- Memory pool management
  - (fixed-sized, variable-sized)
- Time management
  - (cyclic handler, alarm handler, overrun handler)
- System state management
- Interrupt management
- Service call management
- System configuration management

ITRON TCP/IP API Specification

ITRON TCP/IP API Specification suitable for embedded system.

<Approach>

- **Based on the socket interface**
- **Harmonized with the ITRON kernel specification, but can be implemented on other kernels.**

<Differences with the socket interface>

- **TCP API and UDP API are separately defined.**
- **"End point" abstraction is adopted instead of "socket" abstraction.** **TCP end point for waiting for connection requests and TCP communication end point are handled as different objects.**
- **TCP APIs for reducing data copies are also defined.**
- **Non-blocking calls and callbacks are supported.**
- **The callback routine is used for receiving UDP packets.**

JTRON Specification

Standards for communication interface between real-time tasks and Java applications.

<Type1: attach classes>

- Java applications can access real-time OS resources through attach classes.

<Type2: shared object>

- Real-time tasks can access shared objects exported from the Java application
- explicit locking/unlocking mechanism
- Java application must explicitly call the unshared method on the object

<Type3: stream interface>

- Real-time tasks and Java applications can communicate through stream interface.

Debugging Interface Specification

Interface Standard between uITRON-specification kernels and debugging tools, uITRON support becomes easy.

<Scope of the Specification>

- The interface between uITRON-Specification kernels and the RTOS-support functions of debugging tools

***kernel object state reference**

***task-aware breakpoint and stepping**

***kernel trace etc.**

- Goal

***Run-time overhead should be minimal**

***Most part should be common to different kind of debugging tools (debugger, ICE).**

***The basic concept/architecture should be applicable to other RTOS and software components.**

Introduction of Protection Mechanism

Background:

- Requirements for protection mechanism is emerging to facilitate debugging process and to raise system reliability.

- Protection mechanism is required to secure the system from the software downloaded via network.

- Some overhead for protection is now permissible.

Scope of the standardization:

- extension of uITRON4.0 with access protection mechanism of memory and kernel objects (task, semaphore, etc.).

Standardization process:

- Working group for the standardization started in early 2001.

- First version (Japanese) of the specification is due June 2002.

Standardization Approaches

Three different purposes of protection mechanism:

- facilitating the debugging process

- raising the system reliability

- securing the system from downloaded software

Design Policy:

- covering the above three purposes with one specification

- enabling low overhead implementations

- making the specification simple

Approaches to lowering overhead:

- removing the address translation

- making use of static information for optimizations

Summary

- μITRON real time kernel specification is a de-facto industry standard in Japan.
- Several USA RTOS venders ship μITRON-spec RTOS
- Major results of 2nd phase activities
  - MMU function of μITRON4.0 Real-Time kernel spec.
  - ITRON TCP/IP API Specification
  - JTRON2.1 Specification
  - ITRON debugging interface specification
- ITRON Project Web Site http://www.itron.gr.jp