Process Controllers and Simulators

Topic 1

Introduction to Process Control Systems

Course structure

1. Introduction
2. Architecture of Computer Control Systems
3. Organization and structure of computers for control purposes
4. Basic control algorithms
5. Real-Time software environment
6. SCADA
7. Simulators – general theory
8. Simulators – practical aspects
9. Simulation of distributed objects and control systems
10. Simulators validation
11. Real-Time system improvement using simulation environment

Course structure

Chapter		Week														
		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1	Introduction															
		x														
2	Architecture of Computer Control Systems															
			x													
3	Organization and structure of computers for control purposes															
				x												
4	Basic control algorithms															
					x											
5	Real-Time software environment															
						x	x									
6	SCADA															
								x	x							
7	Simulators – general theory															
										x	x					
8	Simulators – practical aspects															
												x	x			
9	Simulation of distributed objects and control systems															
														x		
10	Simulators validation															
															x	
11	Real-Time system improvement using simulation environment															
																x

Introduction

- A modern society and Industrial Control Systems
- Characterization of Industrial Control Systems
- Software engineering for Industrial Control Systems

Modern society and ICS

History of development

1. **Early digital computers**
   - data acquisition systems;
   - supervisory control systems;
   - dedicated systems using special-purpose equipment

Modern society and ICS

History of development

2. **SSI / MSI minicomputer technology:**

Direct Digital Control

(DDC) systems

Modern society and ICS

History of development

3. **LSI microprocessor technology:**

Decentralized and Distributed Control Systems

Modern society and ICS

History of development

4. **VLSI micro-controller technology:**

Mechanical and electronic devices

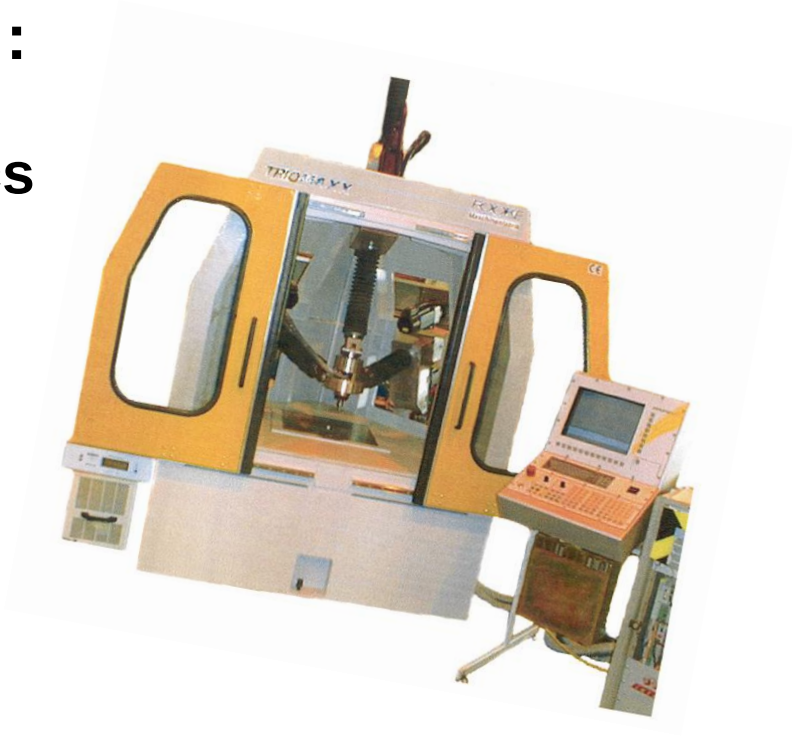
with built-in intelligence

Modern society and ICS

History of development

5. **Integration of control systems and mechanical components :**

   - **Pervasive Mechatronics**



Modern society and ICS

History of development

6. **Integration of communication and control (pervasive computing):**

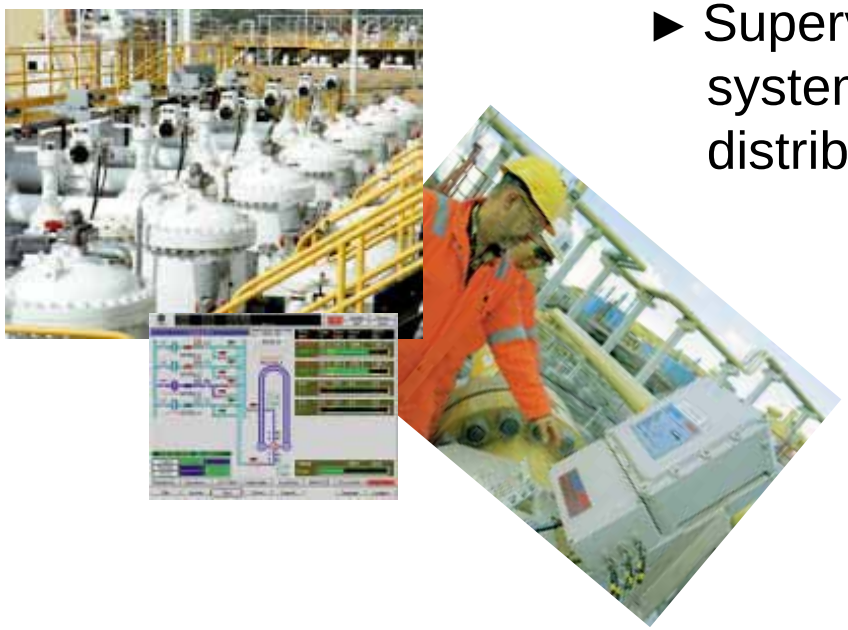
► **Distributed embedded systems (e.g. aerospace and automotive systems)**

► **Integrated plant-wide and company-wide monitoring and control systems**

Modern society and ICS

History of development

6. **Integration of communication and control (pervasive computing):**



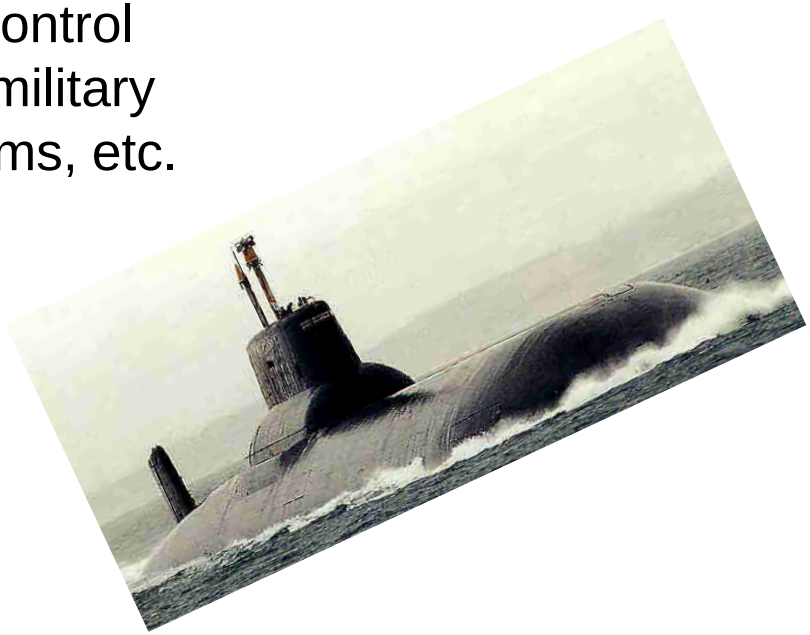
- Supervisory monitoring and control systems (SCADA) - pipelines, water distribution, etc.

Modern society and ICS

History of development

6. **Integration of communication and control (pervasive computing):**

► Large-scale command-and-control systems - air-traffic control, military command-and-control systems, etc.



Modern society and ICS

The post-PC era

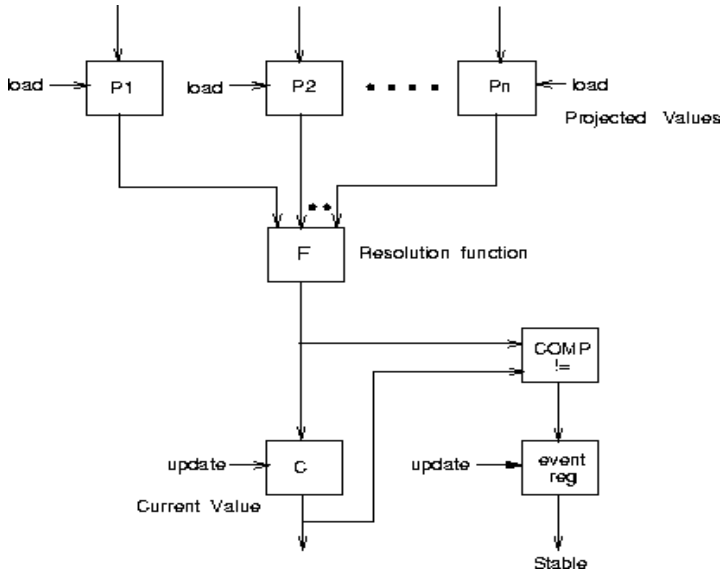
The advent of pervasive computing and pervasive mechatronics.

Modern society and ICS

The post-PC era

Major application areas of modern computer control systems:

- Military applications
- Aerospace systems
- Automotive systems
- Railway transportation
- Sea transportation

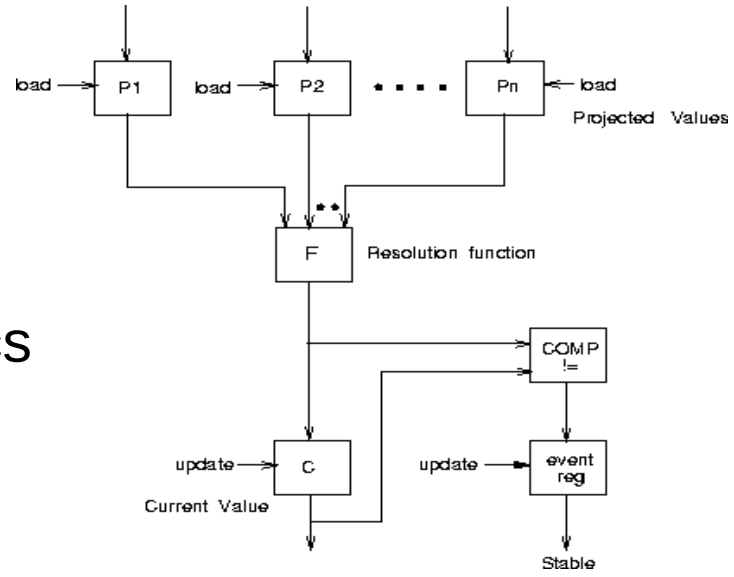


Modern society and ICS

The post-PC era

Major application areas of modern computer control systems:

- Industrial automation
- Medical instrumentation
- Measurement instrumentation
- Materials handling and logistics

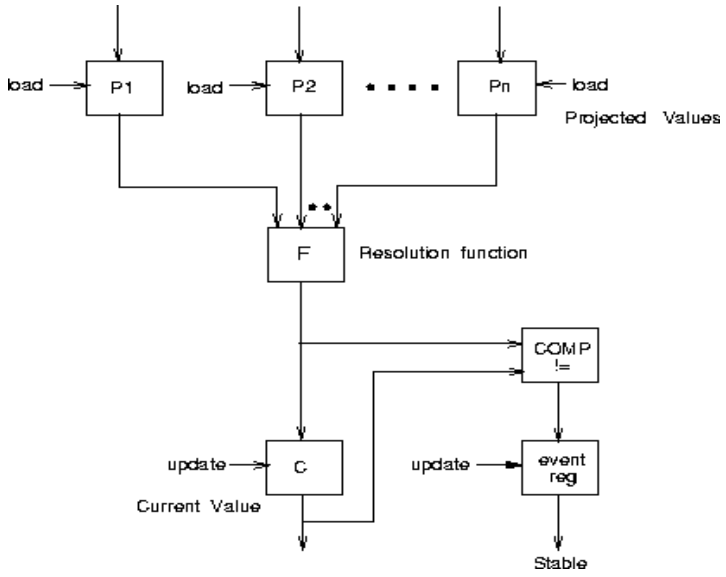


Modern society and ICS

The post-PC era

Major application areas of modern computer control systems:

- Communication systems
- Digital TV and audio
- Home appliances
- Home and office automation



Characterization of ICS

Specific features of ICS

- Reactive systems

  Industrial Control Systems have to be highly reliable in terms of both hardware and software, and to work 100% safe.

- Real-time systems

  Industrial Control Systems have to react with adequate time, of the changes taking place in the objects of control.

- User-friendly systems

  Industrial Control Systems have to be easy to use and operate, and sometimes and their presence must be totally transparent to the user.

- Dependable systems

  Industrial Control Systems have to be adequate in terms of functional correctness and they have to be temporally correct (i.e. before some kind of deadline).

Characterization of ICS

- Scalable architecture supporting both autonomous (stand-alone) and distributed applications
- Reduced development and recurring costs
- Reduced time to market
- Small size
- Low energy consumption
- Low unit cost in volume production
- etc.
- High quality of service
  - Dependable operation through reliable and error-free software
  - Predictable and guaranteed behavior under hard real-time constraints
  - Open architecture featuring reusable components and software reconfiguration, including in-site and on-line reconfiguration .

Classification of ICS

Difference from other computer systems

- The presence of Physical Constrains
  - Temporal Constraints
  - Noise Constraints
  - Synchronization Constraints
  - Dependability Constraints
  - Other physical constrains

The presence of Physical Constrains implies a hypothesis about the world and how the program affects the world.

Classification of ICS

A partial classification of Industrial Control Systems :

- Application functions: data acquisition, monitoring and control systems
- Distribution of functions: stand-alone (centralized) vs. networked and distributed systems
- Domain specific - object of control type of control system: continuous vs. discontinuous (sequential) and hybrid control systems
- Real-time behavior: hard real-time vs. soft real-time systems

Fault tolerance: fail-safe vs. fail-operational systems

State-of-the-art Software Technology

- Ad-hoc design and manual coding techniques, i.e. “manufacturing production” of RT control software
- Computer-aided generation of RT control software from high-level specifications
- Systems and languages supporting limited application / code generation, e.g. standard *IEC 61131-3* and *IEC 61499-3* for industrial applications; special-purpose languages in areas such as aerospace, communications, signal processing, etc.
- Computer-aided code generation from high-level specifications

State-of-the-art software technology

Deficiencies:

- Special-purpose and/or heavyweight solutions that are usually intended for high-end systems and languages.
- Important aspects of real-time operation such as concurrent process execution, process scheduling and schedulability analysis are not adequately supported.
- Code generation techniques are not flexible enough, i.e. they do not provide adequate support for in-site and on-line reconfiguration (compilation and linking of the generated code is required).

Industrial production of ICS software

- Migration from manufacturing to industrial production methods - a historical perspective (e.g., mechanical engineering and electronics)
- Prerequisites:
  - introduction of formal (mathematical) analysis and design methods
  - standardization of components and production methods
  - automation of both design and production

The ultimate solution

Industrial production of ICS software:

Component-based design of software, i.e.

- Migrating from custom design methods and manual coding / testing process to automated methodologies
- Standardization of components and production methods

computer-aided configuration of ICS using formal models (frameworks) and pre-fabricated executable components similar to mechanical and electronics hardware.

- Formal methods for design / analyzes / validation.
- Automation of design / implementation / production processes

Specific problems to be solved

- Formal definition of *comprehensive*, yet *intuitive* and easy to use frameworks
- Formal verification of real-time control systems with respect to *functional* and/or *timing* correctness: the divide-and-conquer approach
- Development of application-specific component libraries
- Development of software configuration/generation and analysis tools
- Development of operational software: *safe* real-time kernels, communication protocols, etc.

The END