Roadmap – Course Overview

- Introduction
  - Cooperation Models and Mechanisms
  - Communication in Cooperative Systems
  - Micro-Level Architectures
  - Macro-Level Structures
  - Development Methods
Introduction

- Basic Definitions and Explanations
  - Cooperation
  - Coordination
  - ...

- A Distributed Systems Perspective
- Examples of Cooperative Applications
- Related Disciplines
“Coordination is the act of working together for a purpose.” (Dictionary)

“Coordination is managing dependencies between activities.” (Malone & Crowston 1994)

“Coordination is a special case of interaction in which agents are aware how they depend on other agents and attempt to adjust their actions appropriately.” (Malone & Crowston 1991)
Cooperation

“Kooperation ist die Zusammenarbeit verschiedener Individuen, die auf der Basis gemeinsamer Regeln (für Kommunikation, Zuständigkeiten usw.) an einem gemeinsamen Ziel arbeiten.” (Dohmen 1994)

key features of cooperation are

- heterogeneity
- regulation
- orientation toward joint goals

agent technology perspective: joint goals have much to do with

- intention to act jointly
- commitment to mutual support, reliable action, etc.
Collaboration and Competition

Introduction • Basic Definitions and Explanations

▶ Collaboration =
  - cooperation *(adopted in this course!)*
  - cooperation among peers
  - cooperation + common problem understanding
  - ...

▶ Competition =
  - coordinated activity
  - conflicting (mutually exclusive) goals
Interaction Types at a Glance

Introduction • Basic Definitions and Explanations
Contrasting Interaction Types

Introduction • Basic Definitions and Explanations

- Coordination vs. cooperation (Grosz 1996):
Contrasting Interaction Types (Cont’d)

Introduction • Basic Definitions and Explanations

► Cooperation vs. sum of individual activities (Grosz 1996):
Interaction vs. cooperation (Grosz 1996):

- Driving in a convoy: a collaboration.

- Driving in Boston: highly interactive, but not a collaboration.
### Components of coordination

<table>
<thead>
<tr>
<th>Components of coordination</th>
<th>Associated coordination processes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Goals</td>
<td>Identifying goals</td>
</tr>
<tr>
<td>Activities</td>
<td>Mapping goals to activities (e.g., goal decomposition)</td>
</tr>
<tr>
<td>Actors</td>
<td>Selecting actors</td>
</tr>
<tr>
<td></td>
<td>Assigning activities to actors</td>
</tr>
<tr>
<td>Interdependencies</td>
<td>“Managing” interdependencies</td>
</tr>
</tbody>
</table>

*(table from (Malone & Crowston 1990))*
Kinds of interdependence

<table>
<thead>
<tr>
<th>Kinds of interdependence</th>
<th>Common object</th>
<th>Example of interdependence in manufacturing</th>
<th>Examples of coordination processes for managing interdependence</th>
</tr>
</thead>
<tbody>
<tr>
<td>Prerequisite</td>
<td>Output of one activity is required by the next activity</td>
<td>Parts must be delivered in time to be used</td>
<td>Ordering activities, moving information from one activity to the next</td>
</tr>
<tr>
<td>Shared resource</td>
<td>Resource required by multiple activities</td>
<td>Two parts installed with a common tool</td>
<td>Allocating resources</td>
</tr>
<tr>
<td>Simultaneity</td>
<td>Time at which more than one activity must occur</td>
<td>Installing two matched parts at the same time</td>
<td>Synchronizing activities</td>
</tr>
</tbody>
</table>
# Processes underlying coordination

<table>
<thead>
<tr>
<th>Process Level</th>
<th>Components</th>
<th>Examples of Generic Processes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Coordination</td>
<td>goals, activities, actors, resources, interdependencies</td>
<td>identifying goals, ordering activities, assigning activities to actors, allocating resources, synchronizing activities</td>
</tr>
<tr>
<td>Group decision-making</td>
<td>goals, actors, alternatives, evaluations, choices</td>
<td>proposing alternatives, evaluating alternatives, making choices (e.g., by authority, consensus, or voting)</td>
</tr>
<tr>
<td>Communication</td>
<td>senders, receivers, messages, languages</td>
<td>establishing common languages, selecting receiver (routing), transporting message (delivering)</td>
</tr>
<tr>
<td>Perception of common objects</td>
<td>actors, objects</td>
<td>seeing same physical objects, accessing shared databases</td>
</tr>
</tbody>
</table>

*(table from Malone & Crowston 1990)*
The Need for Considering Cooperation

Introduction • Basic Definitions and Explanations

- Principle of Bounded Rationality (Simon 1957): the human mind’s processing capacity is limited.
  - The amount of information that can be processed by an individual is limited
  - The detail of control an individual may wield is limited

- Increasing complexity of computer applications (distributed, open, dynamic, etc.)

- AI Perspective: intelligence is not a property of isolated entities (humans, computers), but of “social” entities
  ⇒ to understand intelligence requires to deal with systems being able to interact appropriately
Key features in which cooperative systems differ from one another, and which are suited for characterizing such systems:

- **environment**: diversity, dynamics, predictability, ...
- **cooperating entities**: number, homogeneity, goals, ...
- **cooperation**: frequency, levels, patterns, ...

Space-Time Taxonomy (Weber 1998):

- participants are on the same vs different locations
- participants interact at (nearly) the same vs different time
Introduction

Basic Definitions and Explanations

A Distributed Systems Perspective
- Advantages of Distributed Systems
- Challenges and Requirements

Examples of Cooperative Applications

Related Disciplines
Advantages of Distributed Systems

Introduction • A Distributed Systems Perspective

Compared to mainframes (Weber 1998):

- economic efficiency (cost/performance ratio)
- computational power and speed
- naturally copes with inherent distribution of applications
- reliability
- scalability
Advantages of Distributed Systems (Cont’d)

Compared to pools of independent PCs (Weber 1998):

- resource sharing
- load sharing
- new applications (e.g., teleconferencing)
- reliability
- scalability
Challenges and Requirements

Introduction • A Distributed Systems Perspective

Heterogeneity

networks, computer hardware, operating systems, programming languages, implementations
⇒ middleware (software layer between operating system and applications/services)

Openness

new components – hardware, software – may become part of the system dynamically (during run time), and these components are not necessarily known in advance (at design time)
Challenges and Requirements (Cont’d)

Security
secure access and transmission of information, authentication of users
⇒ denial of service attacks, security of mobile code

Scalability
system remains effective when there is a significant increase in resources and/or users
⇒ control of costs, control of performance, avoid performance bottlenecks, running-out of resources
Challenges and Requirements (Cont’d)

Introduction • A Distributed Systems Perspective

Failure handling
- avoid, detect, tolerate (ignore), recover
- availability

Autonomy and distributed control
- decentralized management control, different management control authorities for different logical and/or physical parts of the system

Mobility
- change in location of information, software, processing, and even hardware
Transparency

distributed system appears as a single virtual system to software entities and end users (hidding of lower-level hard- and software details)

⇒ access transparency (access of local and remote resources with identical operations)
⇒ failure transparency (hidding of failures)
⇒ mobility transparency (user and program operations are not affected by movement of resources)
⇒ location transparency (access of resources without knowing their location)
⇒ concurrency transparency (processes operate concurrently using shared resources without interference between them)
⇒ ...
Other challenges and requirements:

- concurrency
- lack of global state and global clock
- interoperability across platforms and languages
- portability (use of software entities in multiple locations within the distributed system)

NOTE: not all challenges=requirements are (equally) relevant to every distributed system, relevance and appropriateness depends on application (cf. CSCW & transparency)
Introduction

- Basic Definitions and Explanations
- A Distributed Systems Perspective
- Examples of Cooperative Applications
  - Automated Production
  - Docking Station
  - Traffic Flow Regulation
  - ...
- Related Disciplines
Automated Production

Introduction • Examples of Cooperative Applications

- M_11
- M_12
- M_13
- M_21
- M_22
- M_31
- M_32
- M_33
- M_34

Production steps:

A 
B 
C

Parts → Machines → Products
Traffic Flow Regulation

Introduction • Examples of Cooperative Applications
Transportation

Introduction • Examples of Cooperative Applications

X = \{ x_1, \ldots, x_5 \}

Y = \{ y_1, \ldots, y_{10} \}

Z = \{ z_1, z_2 \}

X = \{ x_1, x_2, y_1, y_2, y_3 \}

Y = \{ y_1, \ldots, y_{10} \}

Z = \{ z_1, z_2 \}

X = \{ x_1, x_2, y_1, y_2, y_3 \}

Y = \{ y_1, \ldots, y_{10} \}

Z = \{ z_1, z_2 \}
Robotics

Introduction • Examples of Cooperative Applications
Supply Net/Chain Management

Introduction • Examples of Cooperative Applications

(figure from (Krcmar 2001))
Supply Net/Chain Management (Cont’d)

Introduction • Examples of Cooperative Applications
Meeting

When?
Agenda?
Games – “Blocks World”

Introduction • Examples of Cooperative Applications

agents

A  B  C  D
Games – “Predator-Prey”

Introduction • Examples of Cooperative Applications
Further Typical Application Domains

- telecommunications
- e/m-commerce
- CSCW, groupware
- workflow management
Roadmap – Chapter Details

Introduction

Basic Definitions and Explanations

Examples of Cooperative Applications

Related Disciplines
- Distributed Systems
- CSCW/Groupware
- Workflow
- Systems Engineering
- (Software) Technology
- (Multi-)Agent Technology
Distributed Systems

▶ provides technological foundations of this course
▶ deals with (e.g.)
  ■ networking (principles, protocols, ...)
  ■ interprocess communication
  ■ distributed file servers
  ■ distributed shared memory
  ■ replication, fault tolerance
  ■ name services
  ■ transactions (concurrency control)
  ■ coordination
CSCW/Groupware, Workflow

- concentrates on human-machine cooperation
- deals with (e.g.)
  - meeting support
  - telecooperation
  - group management
  - workflow modeling/monitoring/control
  - coordination and communication
provides methodological foundations of this course

deals with issues such as

- requirements discovery
- modeling (data, processes, system contexts)
- system architectures
- implementation
- maintenance
- testing (validation and verification)
- project management
(Multi-)Agent Technology

- provides conceptual foundations of this course
- deals with “intelligent coordination” from different perspectives, e.g.
  - joint (distributed) planning
  - distributed machine learning
  - distributed decision making
  - negotiation
  - conflict resolution
  - high-level communication