



## EH2750 Computer Applications in Power Systems, Advanced Course.

*Lecture 1 – 3rd September 2013*

*Professor Lars Nordström, Ph.D.  
Dept of Industrial Information & Control systems, KTH  
[larsn@ics.kth.se](mailto:larsn@ics.kth.se)*



## Outline of the lecture

Course Philosophy

Course Administration

Distributed Control in Power Systems

*Reference papers I & II*

Introduction to Multi-Agent Systems

*Course book Ch 1.*




## Course Objectives

- Analyse the needs of advanced functions for power system control and automation
- Document requirements for power system control and automation in a structured manner, e.g. in the form of use cases.
- Independently perform design of advanced functions for power system control and automation based on a set of requirements
- Independently plan and execute a project including requirements analysis and documentation, and design advanced ICT based functions for power system control and automation.
- Implement functions for power system automation and control using predefined components using standardized interfaces.
- Describe the developments in the fields of ICT reliability, security and performance with a specific focus on power system control and operation.

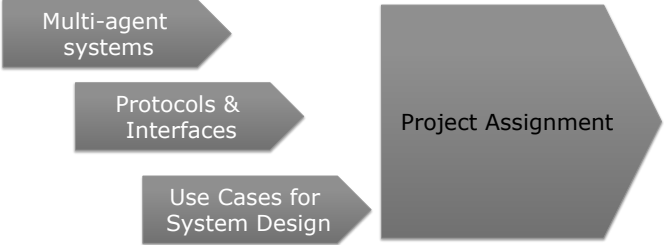


## Other ways of looking at it

- This is a lightweight introduction to Computer Science for Electric Power Engineers
- The course is a programming course
- The course will teach you to work systematically in a group on development of an application from concept to running code.
- The course is an ideal way to identify interesting topics for Master's projects.



EH2750 is a project based course




---




Distributed Control

- Current control systems for power systems are either completely distributed and autonomous (protection) or very centralised (control)
- Development in the Power industry is towards de-regulation and more participating actors.
- In EH2750 we introduce distributed control paradigms utilising "Agents"





---




Protocols & Interfaces

- Control systems are not built from scratch but is instead composed of several components – this is even more true for future applications.
- Interoperability is an over-arching concern in development of computer applications for power systems
- In EH2750 we use real-world interfaces mostly in the right places





---



Use Cases

- To bring some structure to the design of computer applications in power systems the power industry has adopted a Use Case process
- The Use Cases developed are semi-structured
- In EH2750 we bring some order to the chaos




---



## Course expectations

- We offer:
    - A lot of opportunity to work hands-on with developing control applications
    - A structured method to analyse & design applications
    - A portfolio of tools to develop Smart grid applications
    - An open lab with lots of equipment
    - Friendly lab coaches
  - We expect:
    - Creative and positive students
    - A "can-do" attitude
    - Individual responsibility for planning
    - High ethics
- 



## Outline of the lecture

Course Philosophy

Course Administration

Distributed Control in Power Systems  
*Reference papers I & II*

Introduction to Multi-Agent Systems  
*Course book Ch 1.*

---



## Course components

- **Lectures and Exercises**
    - Multi-agent platforms for distributed control of power systems
    - Analysis and design of computer applications in power Systems with use cases
    - Platforms & protocols for computer applications in power systems
  - **Paper Review Assignment**
  - **MAS optimization assignment**
  - **Project Assignment**
- 



## KTH Social


- We will (try to) use KTH Social for communication and administration during the course.
  - KTH Social is reached at [www.kth.se/social](http://www.kth.se/social)
-



### Course Memo Walkthrough

- Course Goals
- Course limitations
- Course contents
  - Lectures & Exercises
  - Paper review assignment
  - Project Assignment
- Course Schedule

[www.kth.se/social](http://www.kth.se/social)



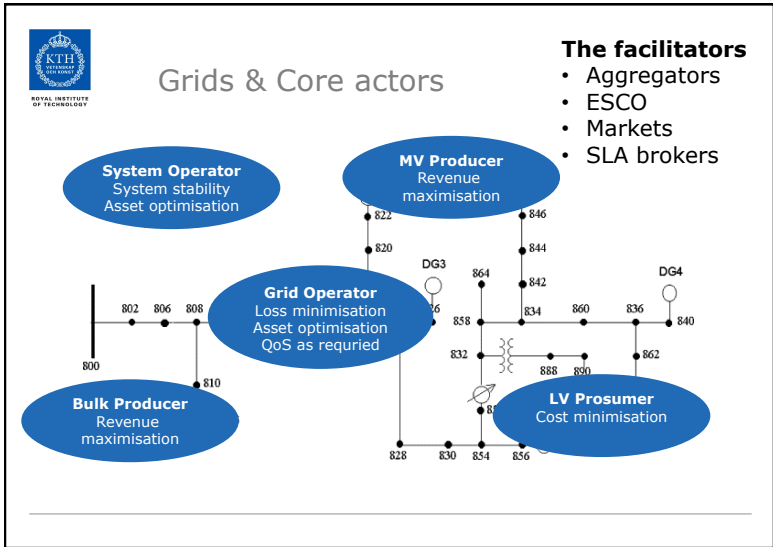
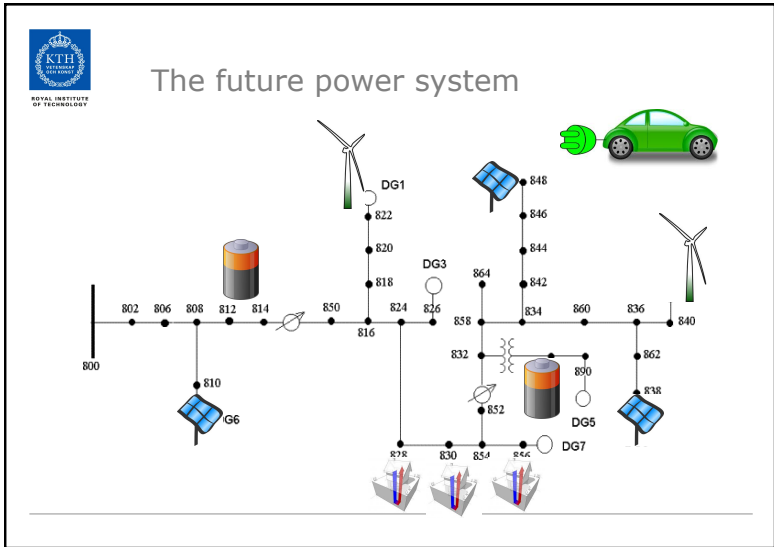
### Outline of the lecture

Course Philosophy

Course Administration

Distributed Control in Power Systems  
*Reference papers I & II*

Introduction to Multi-Agent Systems  
*Course book Ch 1.*




 **New Trends in Power Systems**

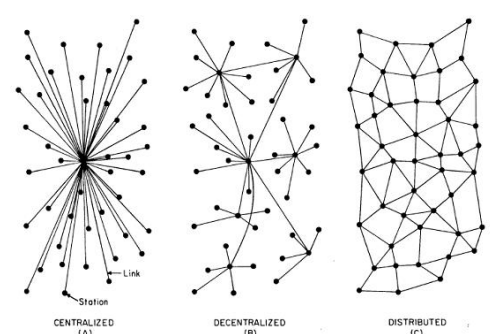
- Deregulation, distributed generation, added actors, business aspects,
- Distributed, decentralized and local control
- Increased complexity and required level of intelligence
- A shift from reactive to proactive mode

- The response/answer is De-centralized control and intelligent distributed control

---


 **Control paradigms**




Station Link

CENTRALIZED (A)      DECENTRALIZED (B)      DISTRIBUTED (C)


---

 **A review of the topic of MAS in Power**



- Comprehensive review of applications of Multi-agent systems in the power engineering domain.
  - Part I – review of work done
  - Part II – Technologies involved in MAS

---

 **When and Where MAS in Power?**

- Requirement for Interaction between distinct conceptual entities, e.g. Different control subsystems
- A very large number of entities must interact
- There is enough data available locally to take decisions and make analysis
- New functionality needs to be added to existing systems
- Overtime there is a need for new functionality to be added

---



## Example of fields of application

1. Monitoring and Diagnostics
    - a) Condition Monitoring
    - b) Post-fault analysis
  2. Distributed Control
  3. Monitoring and Simulation
  4. Protection
- 



## MAS examples within Power

### **IntelliTeam® SG Automatic Restoration System**

IntelliTeam SG is a field-proven universal Smart Grid solution that automatically reconfigures the distribution system after a fault and quickly restores service to segments of the feeder which aren't affected by the fault.



## Outline of the lecture

Course Philosophy

Course Administration


Distributed Control in Power Systems

*Reference papers I & II*

**Introduction to Multi-Agent Systems**

*Course book Ch 1.*


---



## Why Multi-agent systems?


- Five trends run through the history computing:
  - *ubiquity*
  - *interconnection*
  - *intelligence*
  - *delegation* and
  - *human-orientation*

---




## Agents, a Definition

- An agent is a computer system that is capable of *independent* action on behalf of its user or owner (figuring out what needs to be done to satisfy design objectives, rather than constantly being told)





---



## Multiagent Systems, a Definition

- A multiagent system is one that consists of a number of agents, which *interact* with one-another
- In the most general case, agents will be acting on behalf of users with different goals and motivations
- To successfully interact, they will require the ability to *cooperate*, *coordinate*, and *negotiate* with each other, much as people do


---




## Agent Design, Society Design

- The course covers two key problems:
  - How do we build agents capable of independent, autonomous action, so that they can successfully carry out tasks we delegate to them?
  - How do we build agents that are capable of interacting (cooperating, coordinating, negotiating) with other agents in order to successfully carry out those delegated tasks, especially when the other agents cannot be assumed to share the same interests/goals?
- The first problem is *agent design*, the second is *society design* (micro/macro)

Optimization algorithms





The platform for collaboration

---



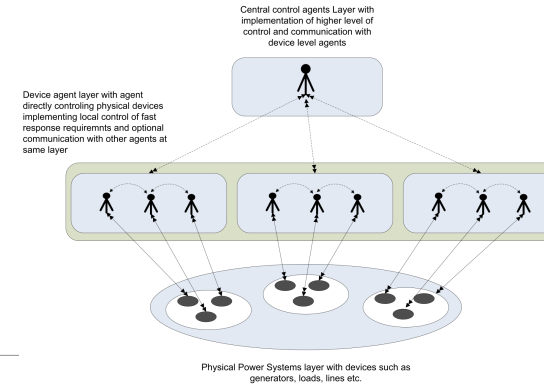
## What is an agent?

An agent is an **encapsulated** computer system that is situated in some environment and can act **flexibly** and **autonomously** in that environment, to meet its design objectives

- Encapsulated
  - Design/Architecture
  - BDI
- Situated
  - Physical and Software (Virtual)
- Autonomous
- Flexible
  - Proactive
  - Social
- Design Objective (Goals or utility)



## Multiagent based distributed control



## Intelligent Agents

- An agent may also be able to:
  - communicate with other software agents
  - learn from experience
  - adapt to changes in the environment
  - make plans
  - reason using, e.g., logic or game theory
  - move between different computers, and/or
  - negotiate with other agents
- Which of these abilities that are implemented in a particular agent depend on its tasks and purpose



## Software vs. Physical Agents

- Software agents
  - situated in a software environment, e.g., operating systems and networks (Internet)
  - *simple example*: software demons (e-mail)
  - *advanced example*: "shopbots"
- Physical agents
  - situated in the physical reality
  - *simple example*: energy saving devices
  - *advanced example*: mobile robots



**Agents vs. traditional programs**

- In expert systems there is a human present between program environment

**Agents vs. Expert Systems**

- Agent reside in the Environment and interacts with it directly (no interface is required)

**Agents are smart and autonomous controllers**

**JACK Architecture**

```

    graph LR
      Agent[Agent] -- has --> Capability[Capability]
      Agent -- post --> Event1[Event]
      Agent -- use --> Plan[Plan]
      Agent -- data member --> BeliefSet[BeliefSet]
      Capability -- handle --> Event2[Event]
      Capability -- use --> Plan
  
```



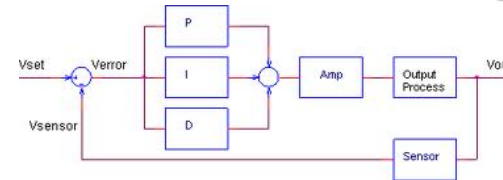
### When to use agents

- Modular - possible to identify distinct "entities"
- Decentralized - entities can perform useful tasks without continuous direction from some other entity
- Changeable - the structure of the system may change quickly and frequently
- Ill-structured - all information is not available when the system is being designed
- Complex - the system needs to exhibit a large number of different behaviors which may interact in sophisticated ways



### But what about a plain old controller?

- Is this not an agent?



- Yes and no, the agent paradigm may very well be used to describe a controller, but it is not necessary.



### Outline of the lecture

Course Philosophy

Course Administration

Distributed Control in Power Systems

*Reference papers I & II*

Introduction to Multi-Agent Systems

*Course book Ch 1.*



Backup slides



## Agents Vs Objects

- **Agents natural extension \evolution of Objects BUT with some very fundamental differences**

- **Level of Autonomy**
  - **Stronger design metaphor**
  - **High Level Interactions – Supporting Organizational structure**
  - **Proactively**
  - **Separate Thread of Execution – Independent life span**
- 



## Agents vs Objects

- It is about adding new abstraction entities:
    - OOP = structured programming + objects that have persistent local states
    - AOP = OOP + agents that have an independent execution thread + pro-activity + greater level of autonomy over itself
  - An agent is able to act in a *goal-directed* fashion rather than just passively reacting to procedure calls
    - “An agent can say no!”
- 



## Agents Vs Objects

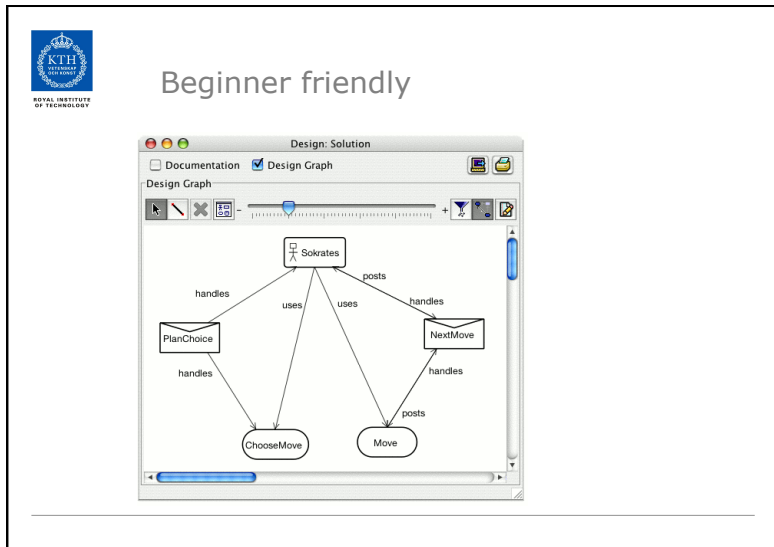
- It is about adding new abstraction entities:
    - OOP = structured programming + objects that have persistent local states
    - AOP = OOP + agents that have an independent execution thread + pro-activity + greater level of autonomy over itself
  - An agent is able to act in a *goal-directed* fashion rather than just passively reacting to procedure calls
    - “An agent can say no!”
- 



## What is JACK

*JACK Intelligent Agents* is an *environment* for building, running and integrating commercial *Java-based* multi-agent software using a *component-based* approach.

---



**JACK Development Environment [by Agent Oriented Software]**

```

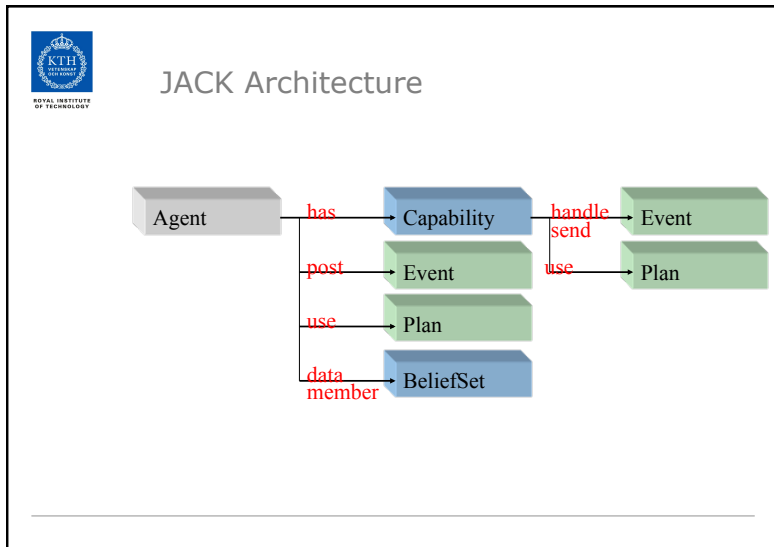
package robot;

public agent Robot extends Agent {
    #has capability Painting painting_cap;
    #posts event Paint pp;

    String paintColour = "black";

    public Robot(String name)
    {
        super(name);
    }

    public void setColour(String c)
    {
        paintColour = c;
    }
}
    
```



- 
- Multiagent Systems in Power Systems**
- In Multiagent Systems, we address questions such as:
    - How can cooperation emerge in societies of self-interested agents?
    - What kinds of languages can agents use to communicate?
    - How can self-interested agents recognize conflict, and how can they (nevertheless) reach agreement?
    - How can autonomous agents coordinate their activities so as to cooperatively achieve goals?