Attendance 10%

CW 30%

Exam 60%

# PROCESS MANAGEMENT

INTRODUCTION

DAYOU LI, PROFESSOR OF ROBOTICS

UNIVERSITY OF BEDFORDSHIRE

UK

## COURSE (MODULE) INFORMATION

- MSc course (of software engineering)
- 32 hours in about 2 weeks
  - Week 16 –
    - Monday to Friday 3 hours from 7 pm to 10 pm (15 hours) plus
    - Sunday from 2 pm to 10 pm (7 hours)
  - Week 17 –
    - Monday and Tuesday from 7 pm to 10 pm (6 hours) plus
    - Wednesday and Thursday 7 pm to 9 pm (4 hours)
- 1 course work and 1 exam

## COURSE INFORMATION 4312

- **Description**
  - This module aims at the knowledge gain (?) of popular process management models such as CMMI, Agile, Lean Six Sigma. You will also develop your capabilities and skills of investigating (critical analysis !) and improving typical software processes.
- **Topics**
  - Processes
  - Model-based process improvement for organisations
  - Capability maturity models integration (CMMI)
  - The use of CMMI
  - 6-sigma
  - PMBOK

## CMMI PRELIMINARY

- Process (a sort of definition)
  - A series actions or steps taken to achieve a particular end
  - In SE, a software development process is the process of dividing software development work into distinct phases to improve design, product and project management
  - Software development life cycle (SDLC) is the process of creating and maintaining software products, and models and methodologies that are used to develop the development and maintenance activities
- Work flow to get job done
  - Work flow itself contains a series steps, such as a flowchart
  - Decision rules applied to the diamonds
  - Forms (interaction interface) to collect clients’ input and to set up target points
  - Dashboard for evaluation and discussion

## CMMI PRELIMINARY

- Key issues in process management or improvement
  - Good practice
  - Standardisation
- Two “legs”
  - Business
    - Establish and maintain business plans
    - Commitment to the plans
    - Cost, schedule and product estimates
    - Evaluation actual work again plans
    - Supply chaining
  - Technical matters
    - Software life cycle from user requirement to maintenance
    - Planned and verified

## PLANNING

1 - 2 - 3 - 4

- Action schema
  - Schema = Precondition + action + postcpndition
  - Action – what you are going to do
  - Precondition – conditions under which you are going to take the action (they are "sufficient" for the action)
  - Postcondition – consequences of the action you take (they are "necessary" for the action)
- Plan for a process
  - A sequence of action schemata each for a step of a process
  - The precondition of an action is the postcondition of the previous action
  - The postcondition of an action is used as the precondition of the successive action
- Elements for planning
  - A set of action schemata
  - A goal (as the necessary condition of the entire plan)
  - The initial situation (as the sufficient condition of the entire plan)

## PLANNING

- Modelling planning
- Predicate logic
  - In strict logic, a prime predicate represents the feature of an object or the relationship between two objects
  - In planning, it represents an action or a condition
  - Prime predicates can be connected together for form compound predicates
- Modelling using predicates
  - Example: Flying a cargo from Airport A to Airport B, with the initial condition that both the flight and the cargo are at A, and both at B as the goal

# PLANNING

- Flying cargo example
  - Actions
    - load(c, p, a) //“loading cargo c onto airplane p that is at airport a”

      PRECOND: At(c,a) $\wedge$ At(p,a) $\wedge$ Cargo(c) $\wedge$ Airport(a) $\wedge$ Plane(p)

      EFFECT: $\neg$At(c,a) $\wedge$ In(c,a)
    - fly(p, from, to) //“flying a airplane from one airport “from” to another “to””

      PRECOND: At(p,from) $\wedge$ Plane(p) $\wedge$ Airport(from) $\wedge$ Airport(to)

      EFFECT: $\neg$ At(p,from) $\wedge$ At(p,to)
  - Initial situation: At(C1, SFO)$\wedge$At(P1, SFO)$\wedge$Cargo(C1)$\wedge$Airport(SOF)$\wedge$Plane(P1))
  - Goal: At(C1, JFK)

## PLANNING

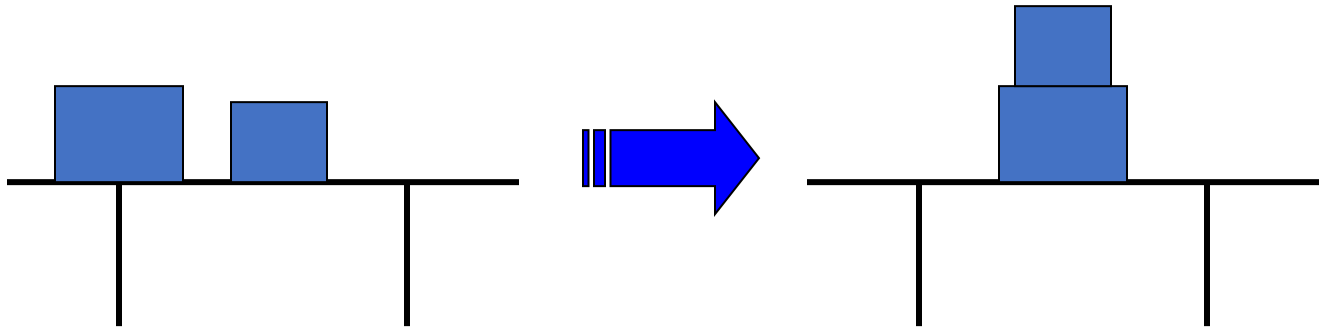
- Planning strategies
  - Forward chaining reasoning
    1. Starting from the initial situation
    2. Comparing it with the preconditions of the action schemata and picking up the schema the precondition of which is the same as the initial situation as the first action in the plan
    3. Comparing the postcondition with the goal, stop if they are the same, otherwise, continue
    4. Comparing the postcondition of that action with the preconditions of the rest action schemata and picking up the schema the precondition of which is the same as the current situation as the next action in the plan
    5. Repeating 3 and 4 until reaching the goal

# PLANNING

- Planning strategies
  - Backward chaining reasoning
    1. Starting from the goal
    2. Comparing it with the postconditions of the action schemata and picking up the schema the postcondition of which is the same as the goal as the first action in the plan
    3. Setting up the precondition of the action as a subgoal
    4. Comparing the subgoal with the initial situation, stop, if they are the same, otherwise, continue
    5. Comparing the subgoal with the postconditions of the rest action schemata and picking up the schema the postcondition of which is the same as the subgoal as the next action in the plan
    6. Repeating 3 to 5 until reaching the goal

# PLANNING

- Modelling planning
- Planning graph
  - Containing vertical bars and links from one bar to another
    - Vertical bars with odd numbers represent conditions
    - Those that have even numbers stand for actions
    - Links associate conditions and actions, so the conditions prior to an action are the preconditions of that action and those after that action are the postcondition
  - Example: Make a plan in a “Block world” towards the goal that Block A on Block B, whilst both blocks are on the top of a table initially



# PLANNING

- Block world example
  - Initial situation: $(On(A, Table) \wedge On(B, Table) \wedge Block(A) \wedge Block(B) \wedge Clear(A) \wedge Clear(B))$
  - Goal: $(On(A, B))$
  - Actions:
    - move(b, x, y) //"move b from x to y"

      Precondition: $On(b,x) \wedge Clear(b) \wedge Clear(y) \wedge Block(b) \wedge (b \neq x) \wedge (b \neq y) \wedge (x \neq y)$

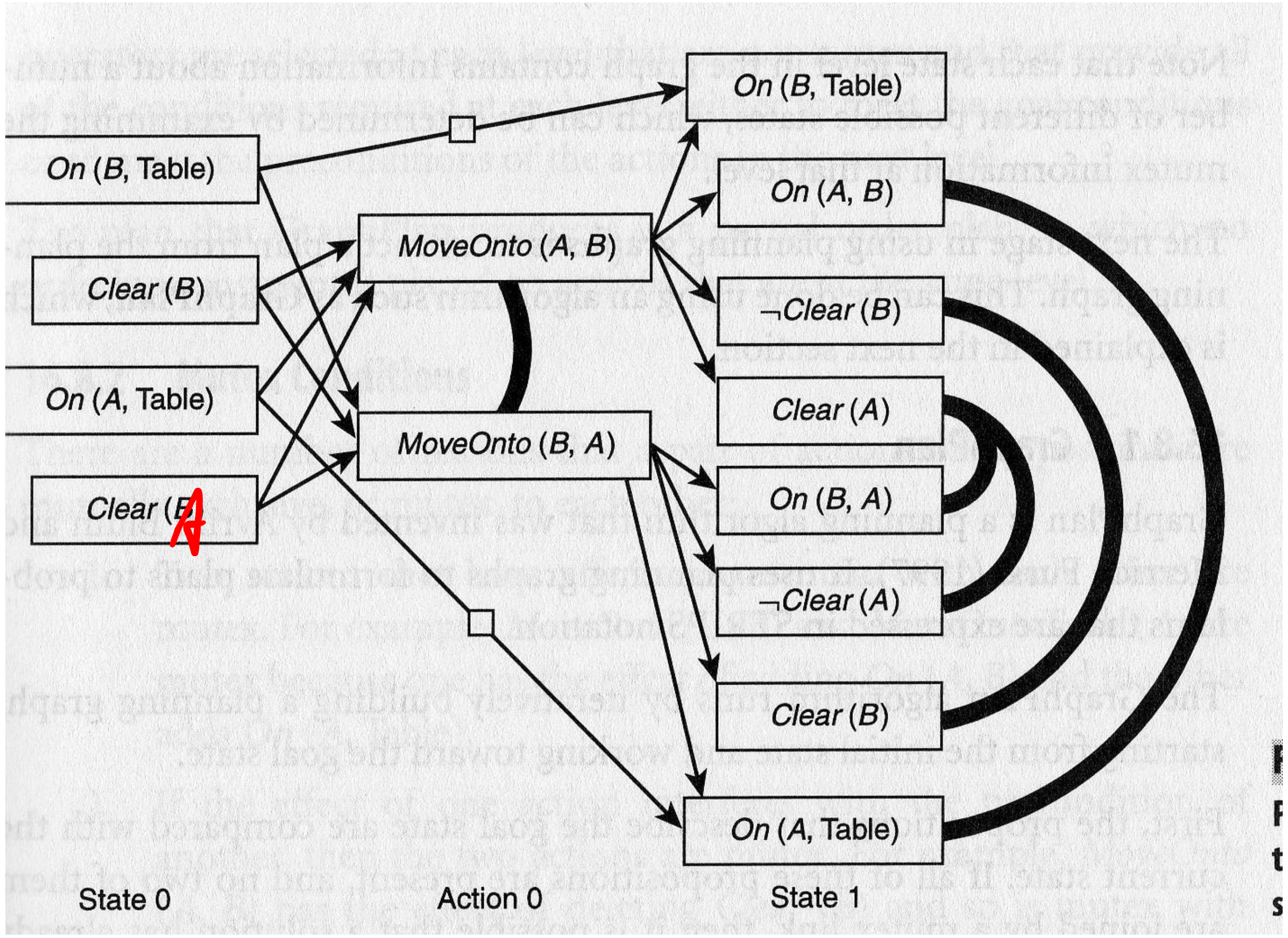
      Postcondition: $On(b,y) \wedge Clear(x) \wedge \neg On(b,x) \wedge \neg Clear(y)$
    - moveToTable(b, x) //"move b from x to table"

      Precondition: $On(b,x) \wedge Clear(b) \wedge Block(b) \wedge (b \neq x)$

      Postcondition: $On(b,Table) \wedge Clear(x) \wedge \neg On(b,x)$

# PLANNING

- Block world
  - Planning graph

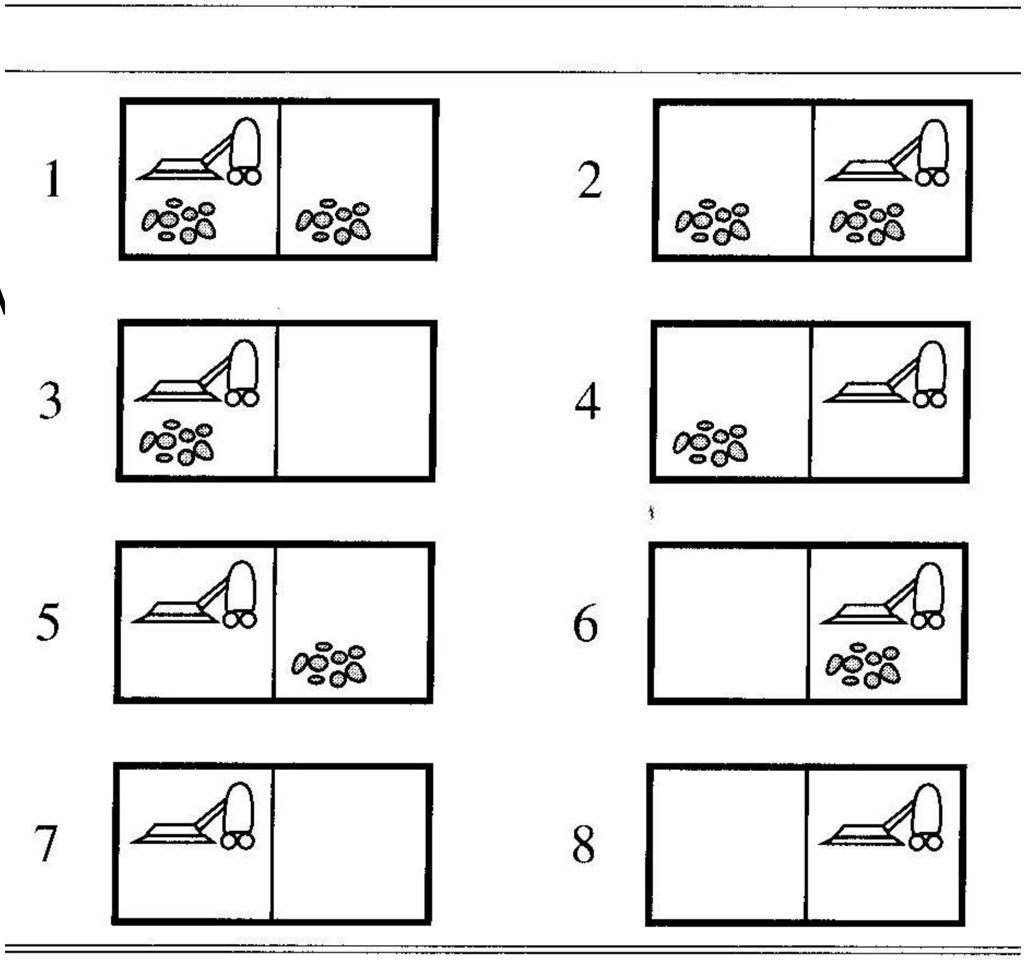


## PLANNING

- Planning with partial information
  - Mental states – believes about situation and environment, and about the believes of other intelligent agents
  - Planning with partial information is based on mental states because of unavailability of information that tells the true situation/environment
  - It sets up mental states as hypothesis and prove or reject them using logical reasoning
  - It produces a set of plans rather than a plan, due to the uncertainty
  - Example: In a “vacuum cleaner’s world”, there are tow room which can be clean and can be dirty, and a vacuum that can suck dirt and move automatically from one room to the other but does not have any sensor.

## PLANNING

- Vacuum cleaner world example
  - The diagram gives 8 physical states
  - Any one or a set can be mental state
  - {1, 3, 5, 7} assumes the cleaner is in A
  - {1, 3} assumes A is dirty
  - {5, 7} assumes A is clean
  - ……..



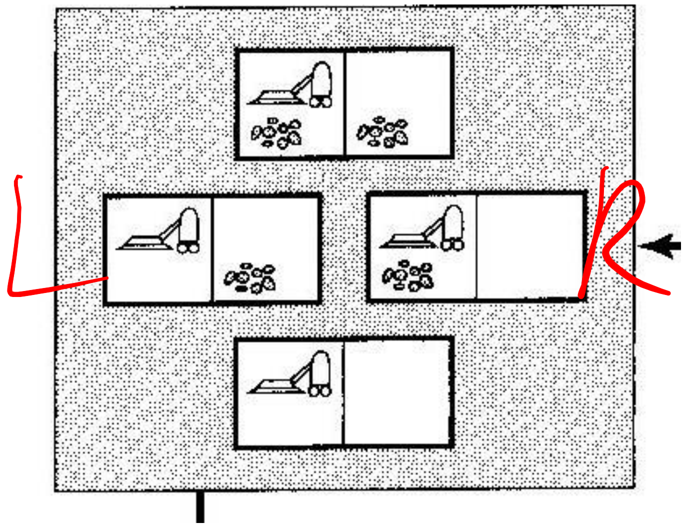
# PLANNING

- Problem modelling
  - Initial belief state is {1,3,5,7} in the example shown in the following diagram provided the agent believes that it is at room A and there may or may not exists dirt in A and/or B
  - Goal state: {8}
  - Actions:
    - Suck, such as

      S:NoDirt (NoDirt is the postcondition of S)
    - Move to the other room, such as

      R: In(roomB) (In(roomB) is the postcondition of R)

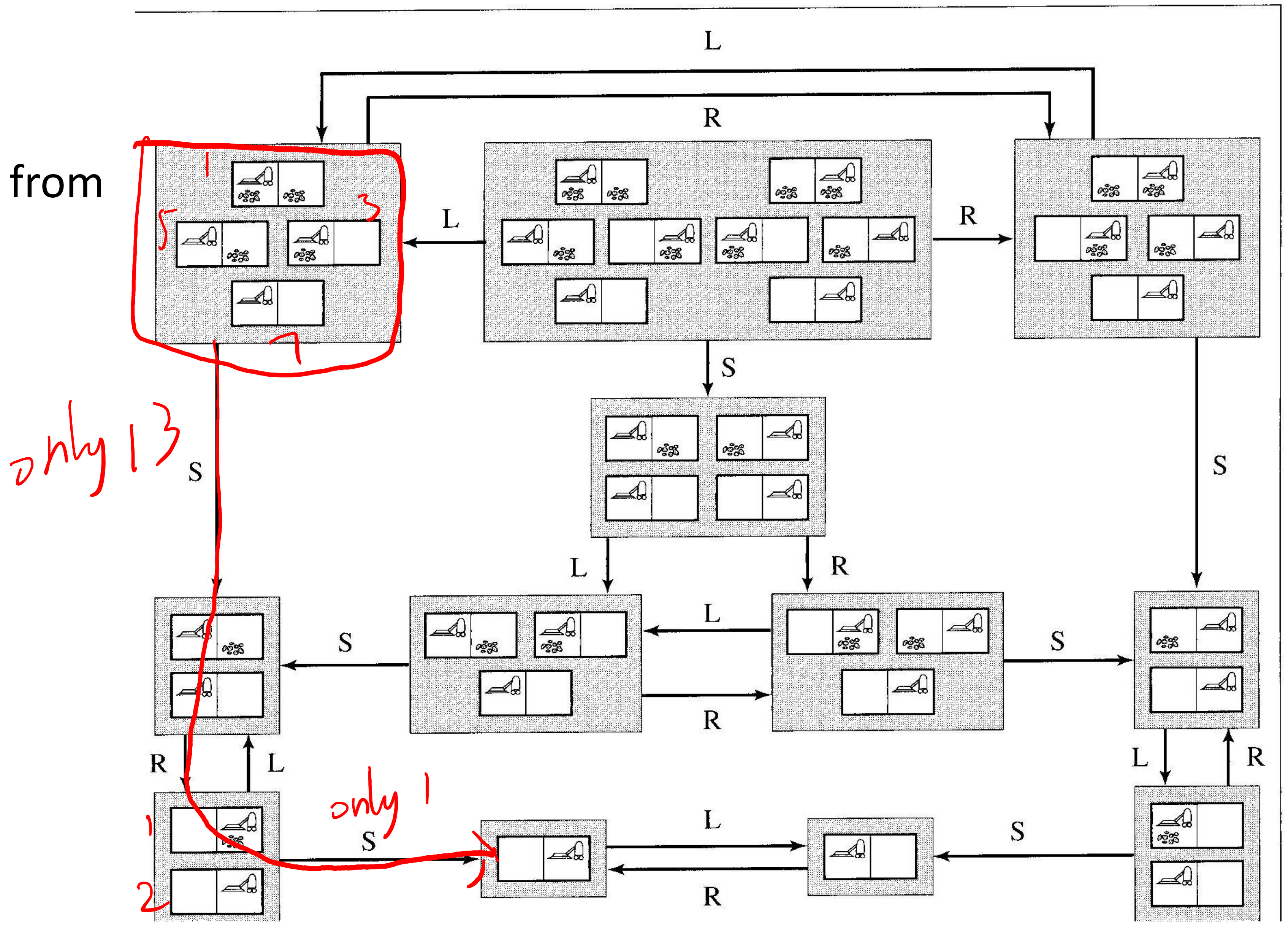
      L: In(roomA) (In(roomA) is the postcondition of L)



# PLANNING

- Plans starting from {1, 3, 5, 7}:
  - S, R, S
  - S, R, L, R, S

x

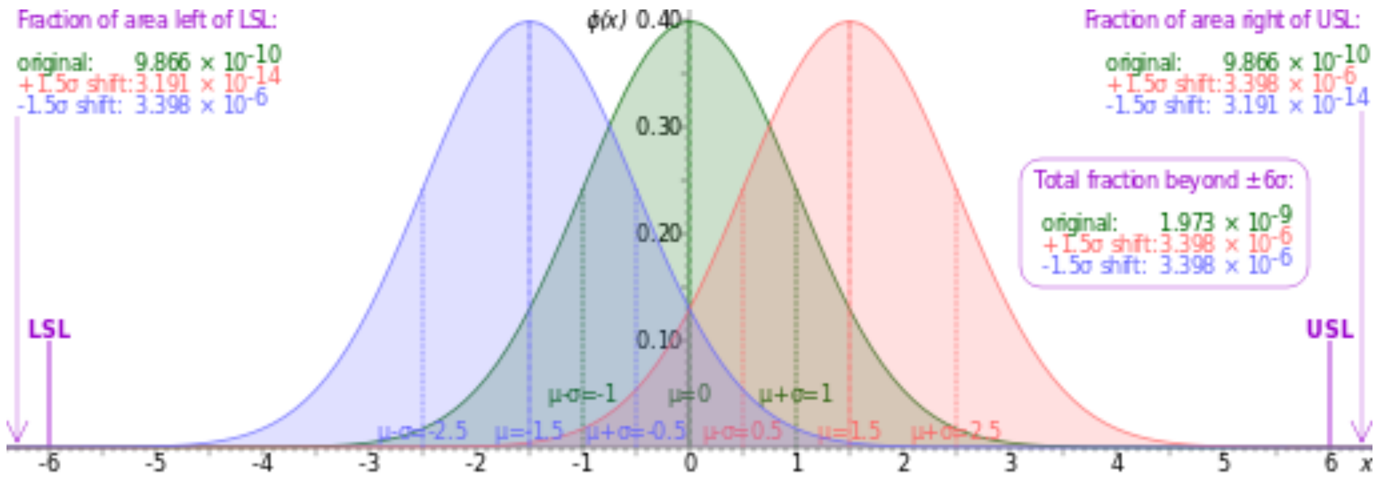


# CMMI PRELIMINARY

- Traditional process management
  - Total Quality Management (TQM)
    - The ”father” of more recent process improvement models such as
      - Six Sigma, Lean Management, Lean Six Sigma, Agile Management, Re-engineering, Total Quality Management, Just-In-Time, Kaizen, Hoshin Planning, Poka-Yoka, Design of Experiments, and Process Excellence
    - The philosophy of TQM is
      - Managerial responsibility for continuous improvement (ownership)
      - Focus on the work processes to achieve improvements (system)
      - Use of statistics to measure process performance (data)
      - Employee involvement and empowerment (invest in people)

# CMMI PRELIMINARY

- Traditional process management
  - 6-sigma ($6\sigma$)



  - Connection between which sigma levels a company is operating at and the cost of poor quality
    - a company operating at a level between three and four sigma spends about 25-40 per cent of their annual turnover taking care of problems
    - a company operating at a higher Six Sigma level only spends about five per cent of the turnover.

## CMMI PRELIMINARY

- Traditional process management
  - Just-in-time (JIT) is an inventory policy of “to order exactly what is needed at the exact time it is needed"
    - Quality -- the exact quantities delivered will all be good enough for use
    - Time -- schedules be maintained exactly as planned and all the support systems come through successfully
    - The implementation of JIT requires continuous process improvement to reduce waste of resources such as
      - Extra inventories
      - Extra capacities
      - Extra time

# CMMI PRELIMINARY

- Traditional process management
  - Lean is also based on the philosophy of eliminating waste
    - Lean organizational structure of a company eliminates many levels of management, bringing everyone closer to the processes
    - Strong process analysis orientation in terms of evaluating every step in the work processes to recreate processes at the steps that add value and to reduce or eliminate those that don't
  - Theory of constraints (TOC)
    - Based on the idea of managing the bottlenecks in a process, known as the constraints
    - Exposing the problem in a process and trying to exploit it

# CMMI PRELIMINARY

- Traditional process management
  - CMM – model-based process improvement
    - C – Capability of completing processes
    - M – Maturity, by improving its process capabilities, the organisation matures
    - M – Models, (what) sets of standardised good practice that can guide the organisation in its processes and in standardisation of good practice, (why) models can also be modified to increase the organisation’s capabilities so it matures
    - (how) Management commitment and an appraisal

## COURSE WORK

- Identify a business or a technical process in your company
- Describe how the process is managed (practice), from planning to evaluation
- Point out any process improvement activities (did you modify the practice?)
- Describe the ”philosophy” that is used to guide the process improvement
- Describe the subsequence of the process improvement (do you feel your company become maturer after the improvement, e.g. better product quality, less waste, etc.?)
- Was any standard generated?