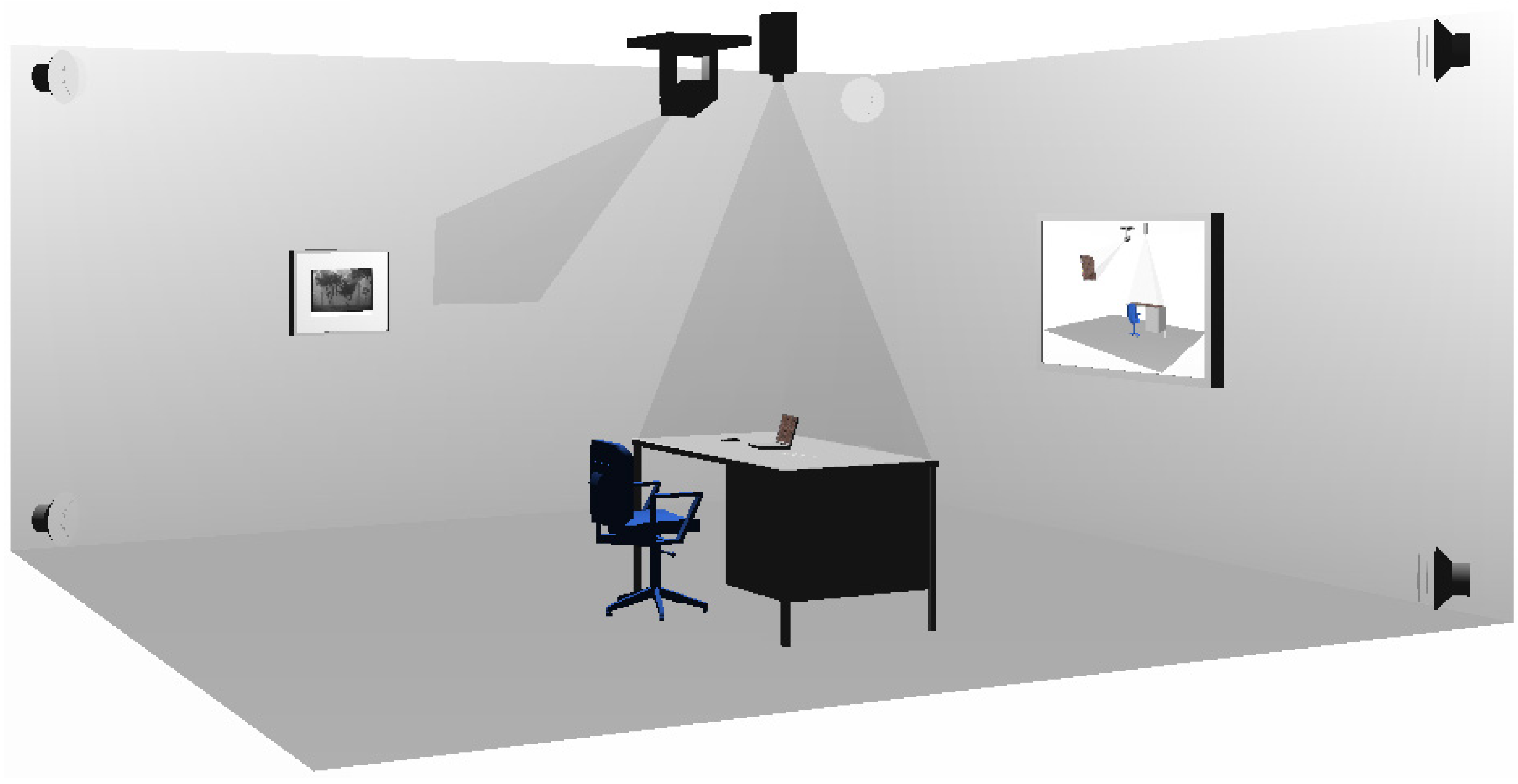
Instrumented Environments

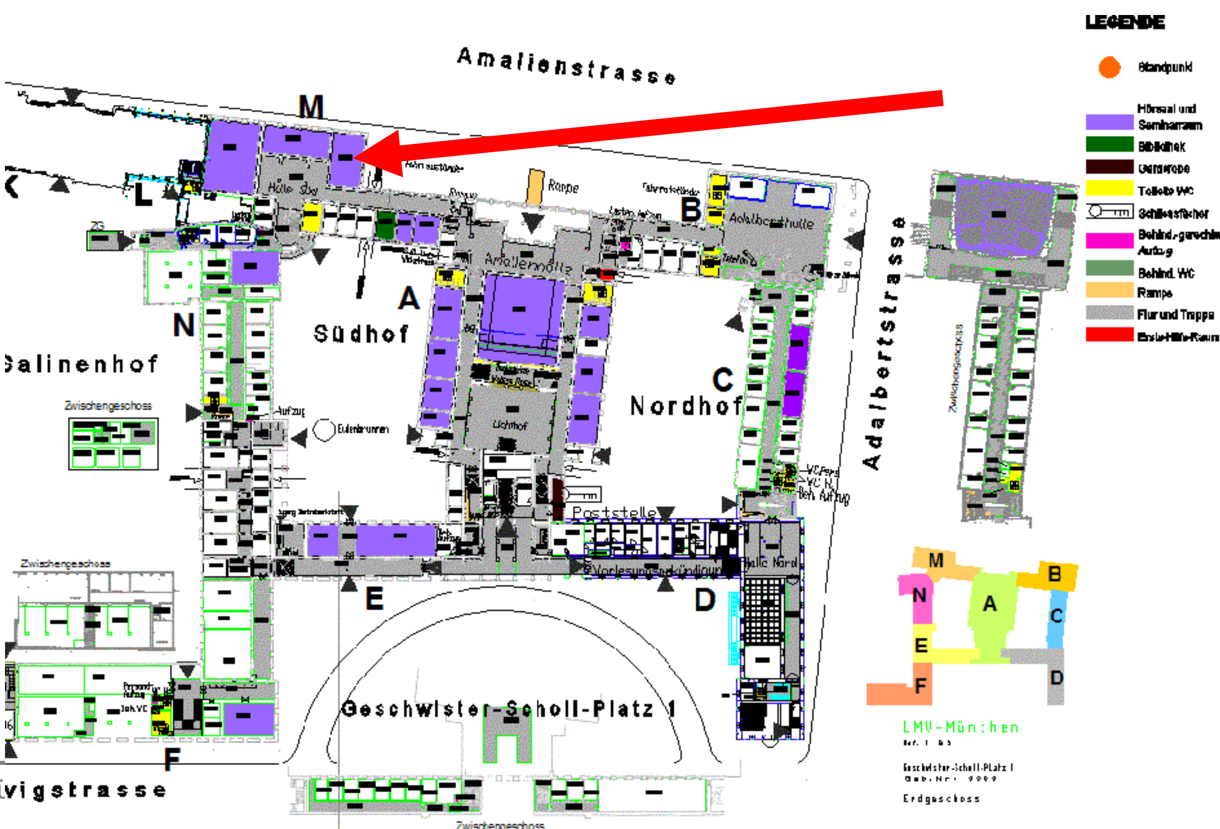
Andreas Butz, butz@ifi.lmu.de, www.mimuc.de

Mon, 10-12 Uhr, Theresienstr. 39, Room E 46



Tomorrow: lecture by Marc Böhlen

- **Machines for Supermodernity**
- Dienstag 14.12.04, 12:15 Uhr
  LMU Hauptgebäude, Raum 129 / M010
- Abstract at
  www.mimuc.de



Topics Today

- Context awareness
  - Some implementation concepts
- Intelligent instrumented environments
  - Knowledge representation
  - Reasoning with this knowledge

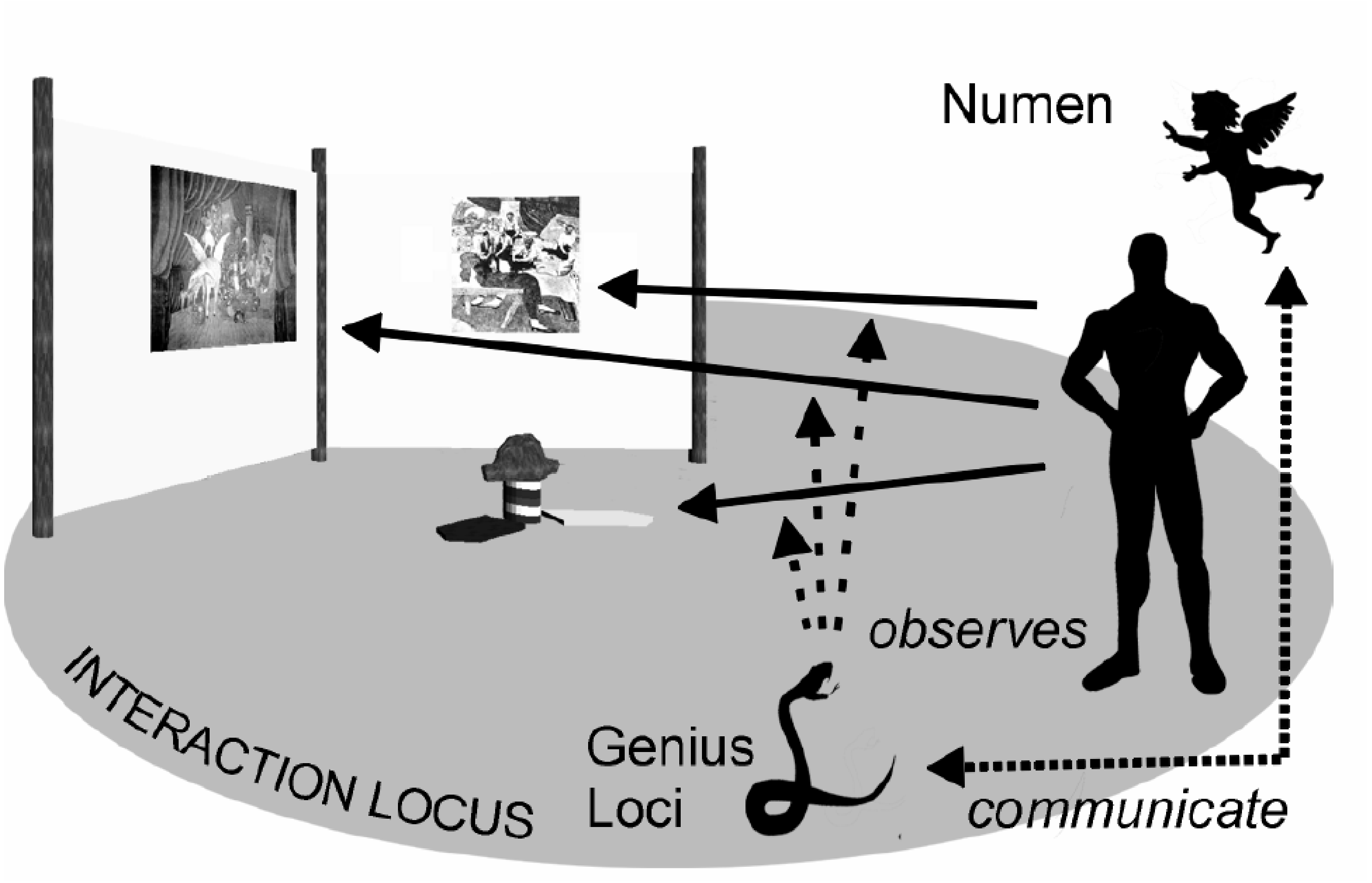
Distributed context aware interaction
(Celentano et al. 2002, 2003)

- Problem: interaction in large intelligent environments
- Solution: use agents to structure the problem space
  - Interaction Locus *IL* (3d-space) and the User
  - Agents: Genius loci and User's numen
- Target Scenario: intelligent museum guide

Distributed context aware interaction

- Genius loci:
  - Knows about displays and interaction possibilities of the interaction locus
- User's numen
  - Knows about the interests and profile of the user and his exploration history

Distributed context aware interaction



Distributed context aware interaction

- Communication protocol integrates knowledge and is started when the user enters the *IL*

1. Genius Loci (GL) starts the dialog and contacts the user’s numen (UN)
2. The UN explains the user’s interest and the GL adapts the properties of the interaction
3. The GL learns user preferences and interaction styles
4. When leaving the IL the GL informs the UN about his inferences

Context Shadow: Organizing Contexts
(Jonsson 2002)

- Problem: How can a huge context be structured and managed?
- Solutions: Design of a searchable topology
- Provide a JAVA API that provides easy access to sensor and personal information with the help of a blackboard architecture

Context Shadow: Organizing Contexts

- Goals of Context Shadow:

1. Support for context aware service discovery.
2. Organization of services and context information in meaningful collections.
3. Context information for applications derived from sensors and other services
4. Refinement of context information.

Context Shadow: Organizing Contexts

- Entities on the blackboard
  - Context servers (CS) that represent persons, locations and groups
  - CS contain information about context and links to other CS
- Implement CS with the help of TSpaces from IBM

TSpaces (Lehmann et al., 1999, IBM)

- TSpaces is network middleware for ubiquitous computing
- A network communication buffer with database capabilities
- Implemented in JAVA
- Communication via Tuples that are read and written to “spaces”

Cross references with Context Shadow

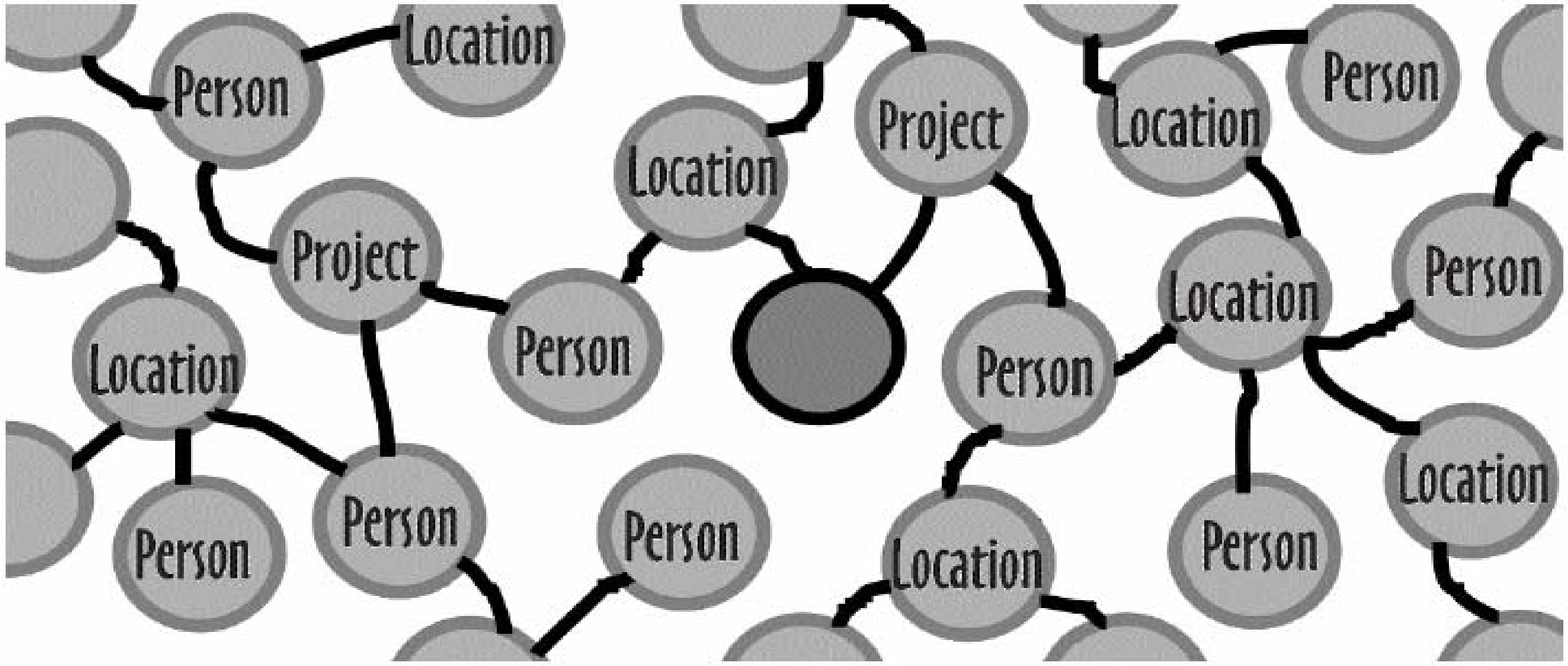


Figure 1. The linked context servers create a searchable space, where the topology of the space is part of the context information.

Example: representing spatial relations

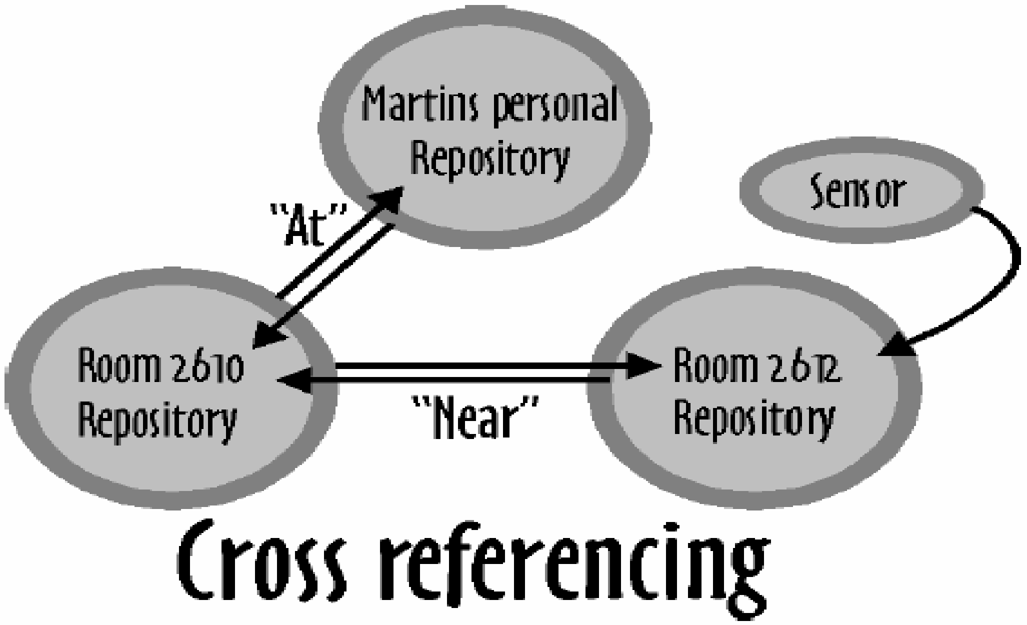


Figure 2. The context servers are linked with references. By following the links it is possible to acquire context information from other context servers than your starting point.

Example: Discover resources

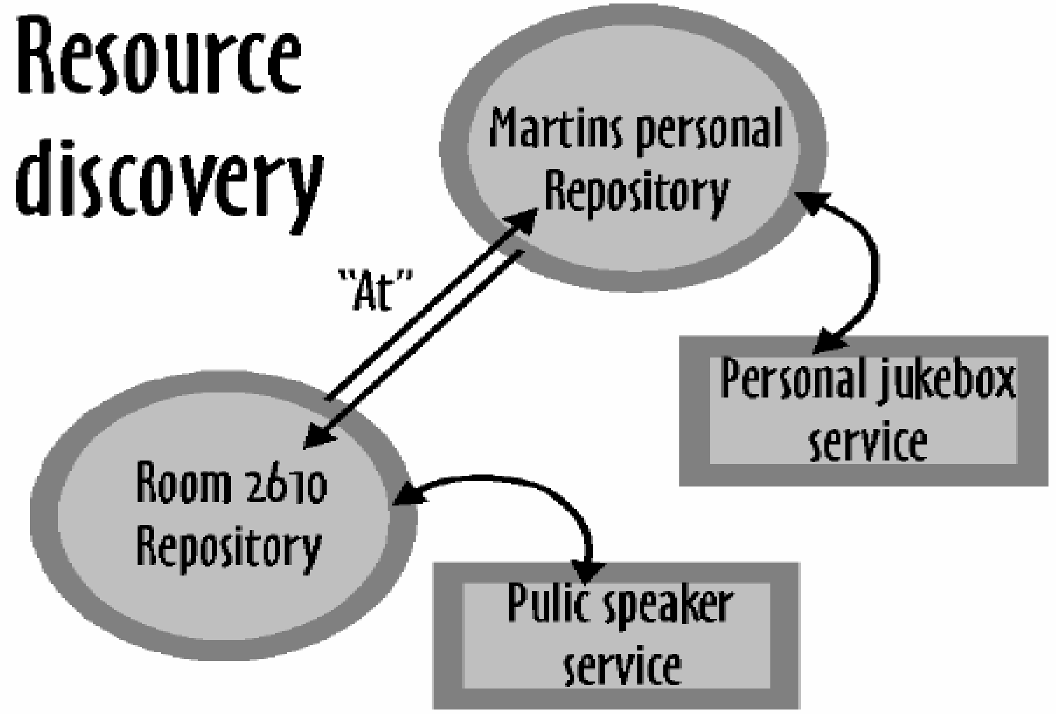


Figure 3. Using Context Shadow, a *jukebox service* finds a *speaker service* at the users current location.

Applications of Context Shadow

- 1. Example: Messenger services
  - Detects resources in the environment
  - Renders messages on appropriate medium
    - Wall displays
    - Public audio
    - Private displays
- 2. Example: Tools for local collaboration
  - Detect JINI services in the environment that are relevant to certain people
  - Support working groups that have gathered by providing last documents

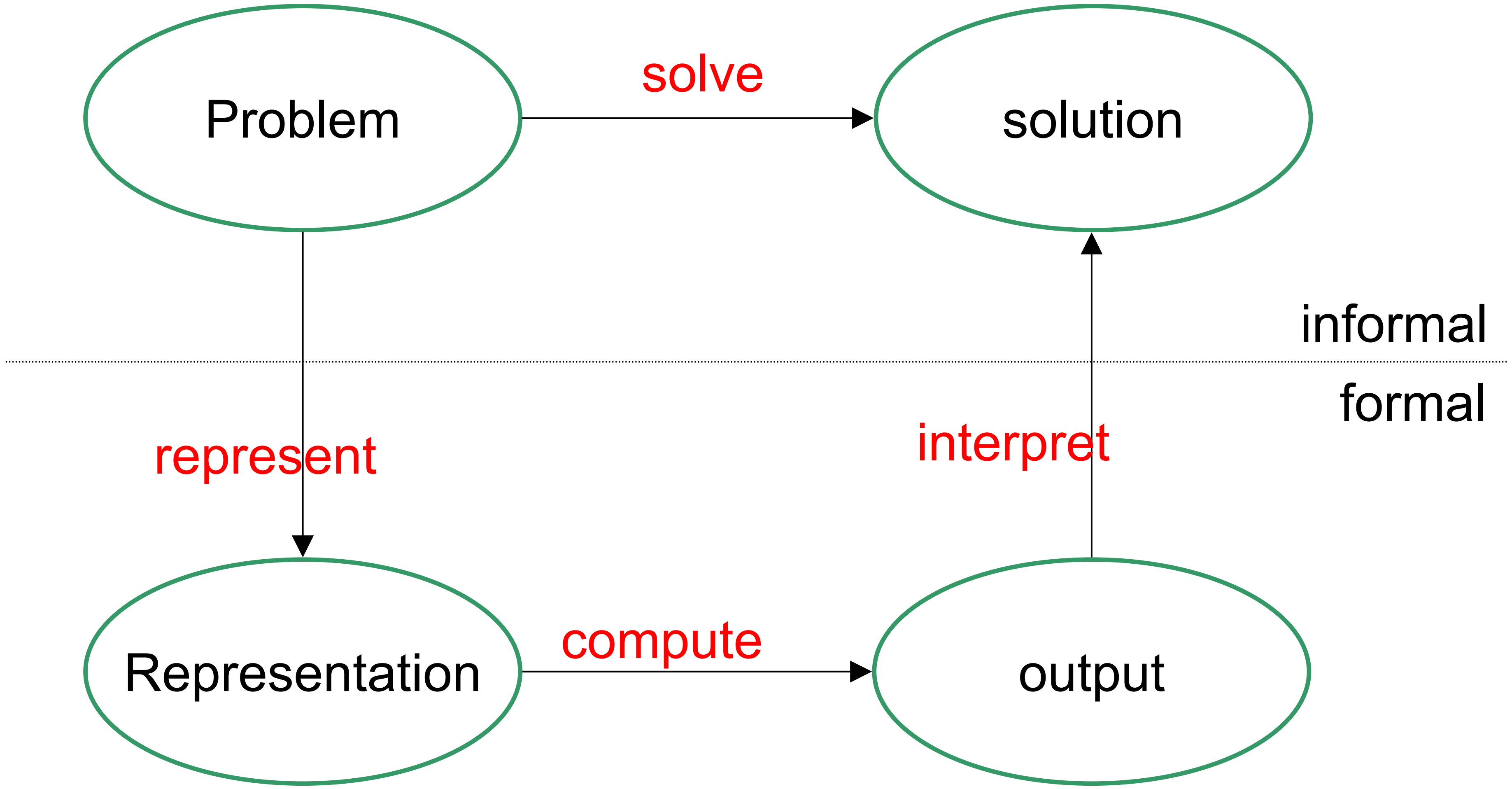
Intelligent Instrumented Environments

- Knowledge representation (KR)
  - Basic concepts
  - Declarative knowledge
    - Semantic networks, frames, production systems
  - Probabilistic knowledge
    - Bayesian networks, FOMDPs
  - KR in instrumented environments
    - Small example

Why KR?

- Real world problems aren‘t formally specified
  - E.g., „Postman should bring packages always on time“
- If a computer has to solve a problem,
  - It must be formalized
  - The solution must be described formally
  - The problem must be represented adequately

Knowledge representation



Basic concepts

- knowledge
- Knowledge representation
- Knowledge representation language
- Knowledge base/source/unit
- Meta knowledge
- Heterogeneous knowledge base
- Multiple representations

Basic questions about formalization

- What is a solution to this problem?
- What properties does a language to describe this problem need to have?
- How can the informal problem be represented in this language?
- What distinctions in the real world are relevant to this problem?
- What knowledge is needed?
- What level of detail is needed?

Basic questions about formalization (2)

- What solution strategies do exist?
- Does the program need to be optimized for *worst-case* or *average-case*?
- Does the user need to be able to understand the solving algorithm?
- Where does the expert knowledge come from?

Defining the solution

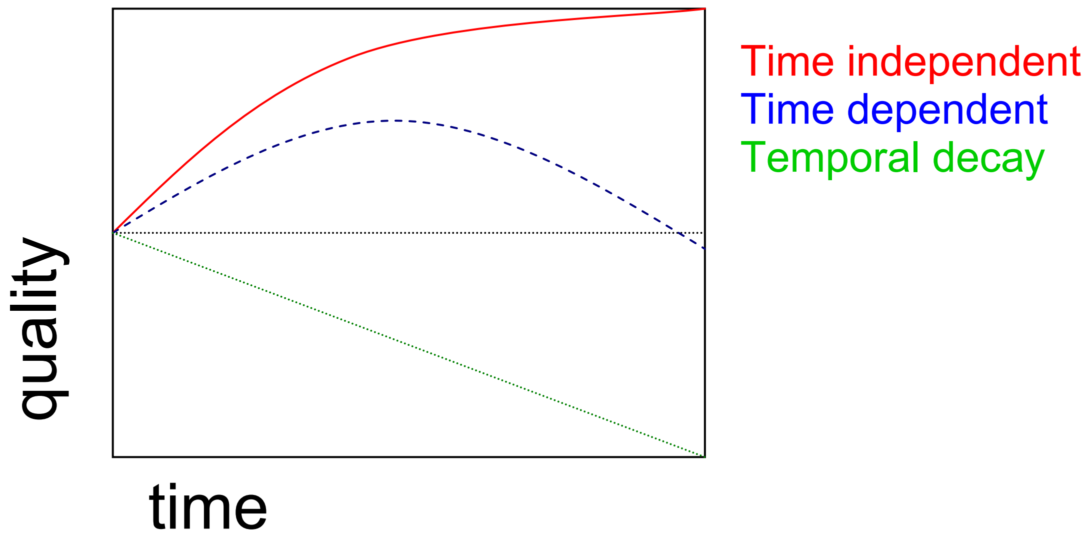
- Main problem: unspecified parts of the problem
  - Example: household robot told to remove all garbage
- One of the biggest AI challenges: formalization of and computing with everyday knowledge
- When a solution was specified: what quality criteria must be fulfilled?

Solution qualities

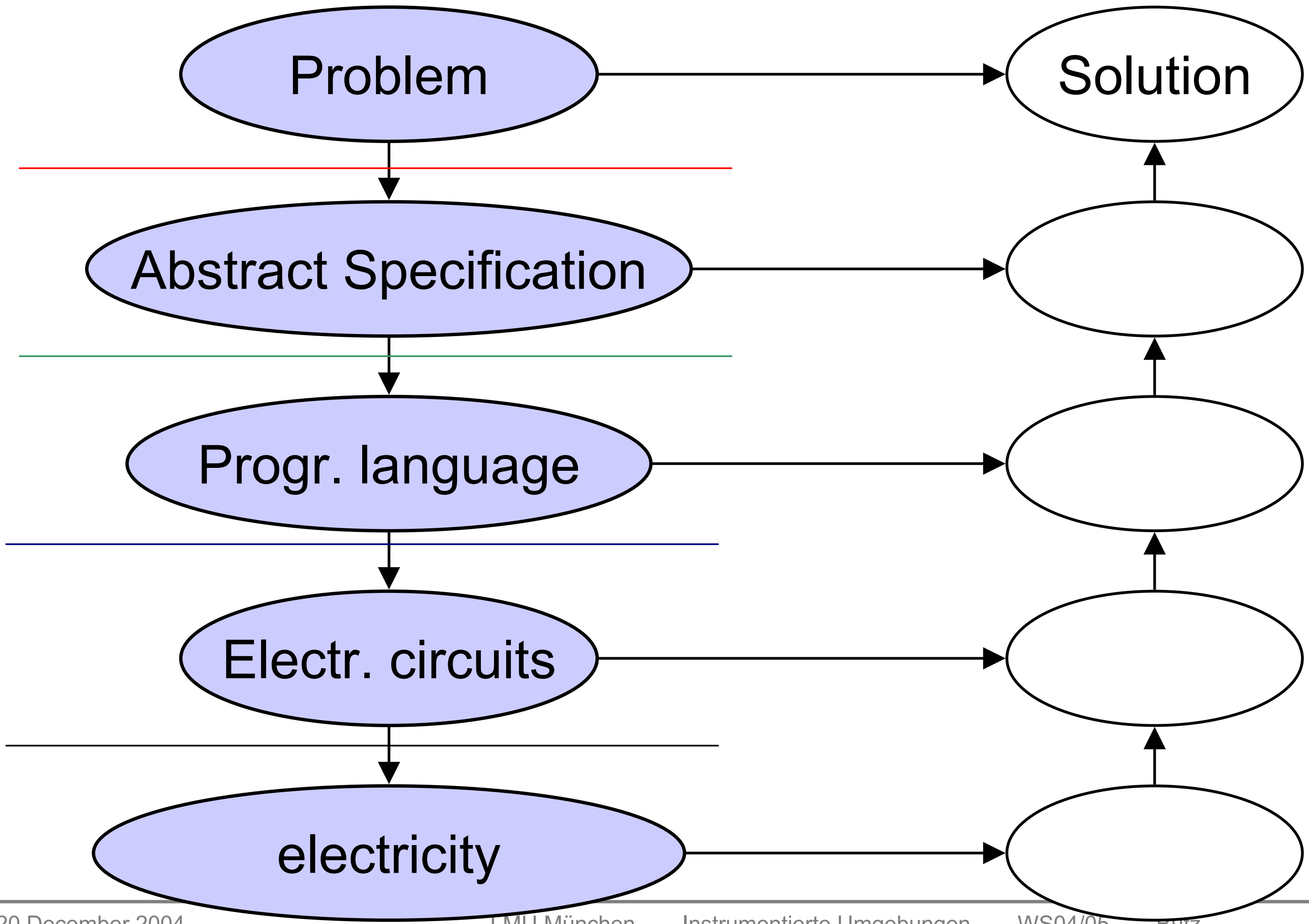
- Finding a solution at all
- Finding a *satisfactory* solution
- Finding an *optimal* solution
- Finding a *probable* solution
- Dealing well with *unexpected* input (robust behavior)

Cost and anytime algorithms

- Computation costs resources (time, space, money…) that should only be invested if the solution is „worth it“
- The result quality of an anytime algorithm increases over time
- An anytime algorithm can be stopped at any time



Levels of representation



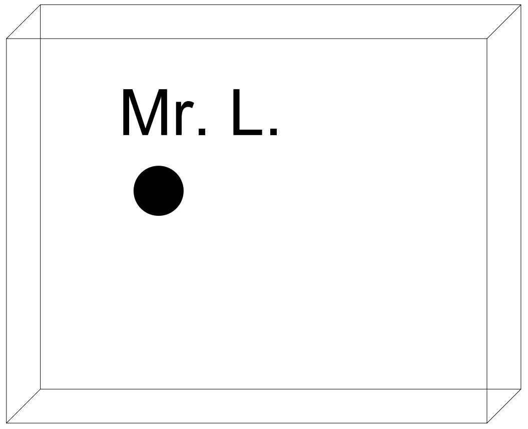
Levels of KR

- Implementation level
  - Atoms, Pointers......
- Logical level
  - Predicates, Quantifiers.....
- Epistemological/Ontological level
  - How the world is constructed
  - How we perceive the world
  - Inheritance relations, structure
- Conceptual level

Qualitative und Quantitative Representations

- Quantitative Representations are based on numerical values
  - Differential und Integral computation
  - Analog Representations, e.g. 3D-models
- Qualitative Representations are based on Boolean logics
  - Boolean values represent important facts
  - Closer to abstract human way of thinking

Bridge between quantitative and qualitative Representations



Room 1.21

Quantitative:

Mr. L. at Position 124, 147
Room 1.21. Position (0,0,200,200)

↓ semantic Transformation

Qualitative:

in_room(Mr. L., Room 1.21)

Objects and Relations

- Statements about objects and relations between objects (ontologies)

Very simple example:
Predicate red

red(*gift*)	the gift is red
?red(X)	What is red? (X is a variable)

But how do you ask: „what color has the gift?"
?X(*gift*) *is not possible* (Why? :-)

Solution: new Predicate color(obj, value)
?color(gift,*red)* *or* ?color(*gift*,C*)*

Objects and Relations

Additional problem:
How can I ask about properties?

example:
Which property of *gift* has the value *red?*

Solution: object-attribute-value Relation (Proposition)

prop(*gift, color, red*) : gift has the color red

generally: prop(**Object, Attribute, Value**)
e.g.: prop(a,is_a, gift)

Boolean Predicate: prop(a, gift, true)

Objects and Relations

- Transformation of complex Predicates to Propositions:
  - scheduled(C,S,T,R) (section S of course C at time T in room R)
- in Propositional Form with a booking “b1”:
  - prop(*b1, course*, C)
  - prop(*b1, section*, S)
  - prop(*b1, time*, T)
  - prop(*b1, room*, R)

Frames

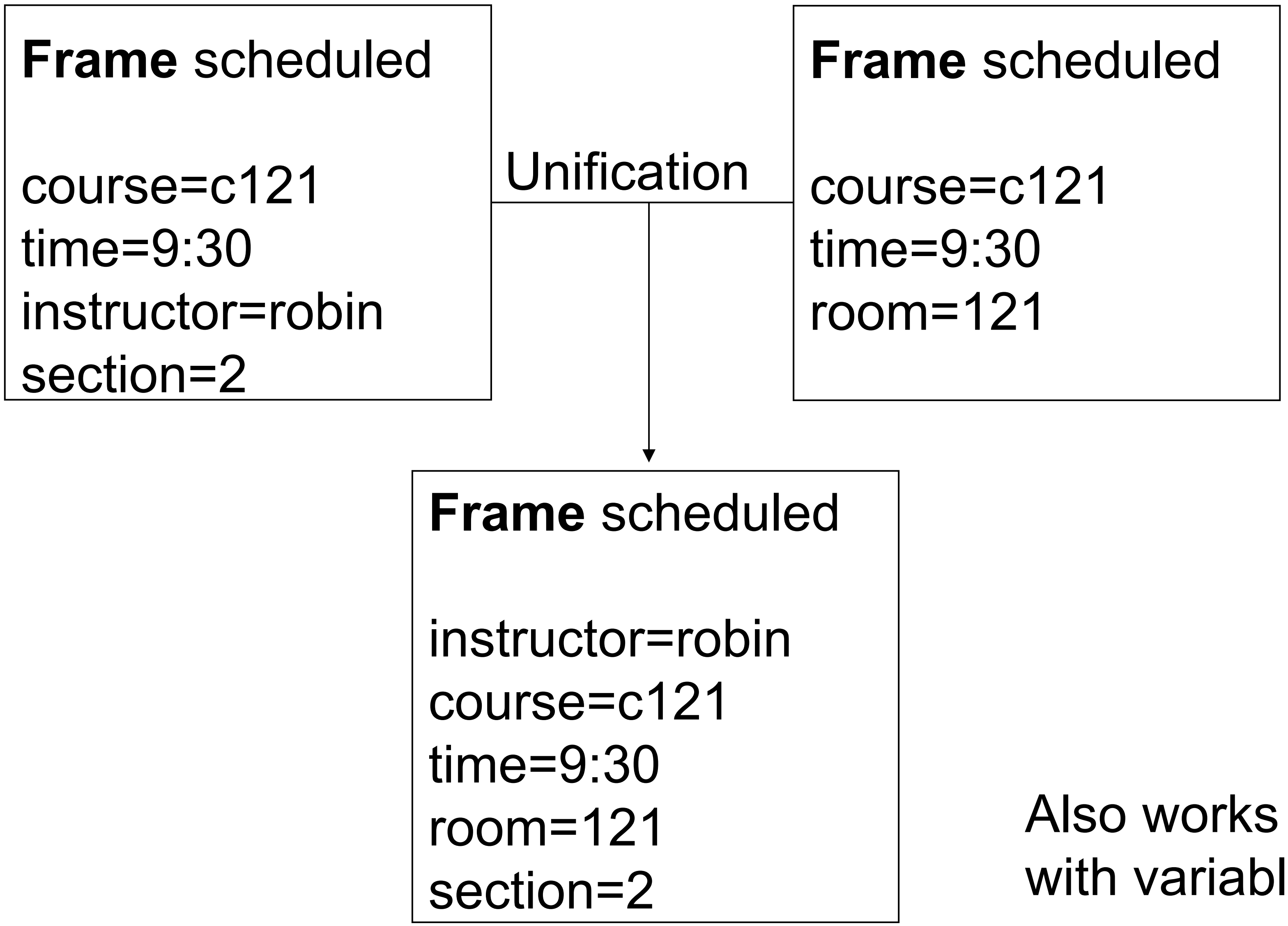
Alternative Representation of knowledge, where all attribute-value Pairs for a specific entity are collected. Attributes are called *slots,* values are called *fillers*

scheduled(C,S,T,R) $\longrightarrow$

Frame scheduled
course=C
Section=S
time=T
room=R

Advantages:
- Instantiation with Defaults
- Slots are position dependant
- Allow Unification
- Inheritance

Unification



Frame scheduled

course=c121
time=9:30
instructor=robin
section=2

Unification

Frame scheduled

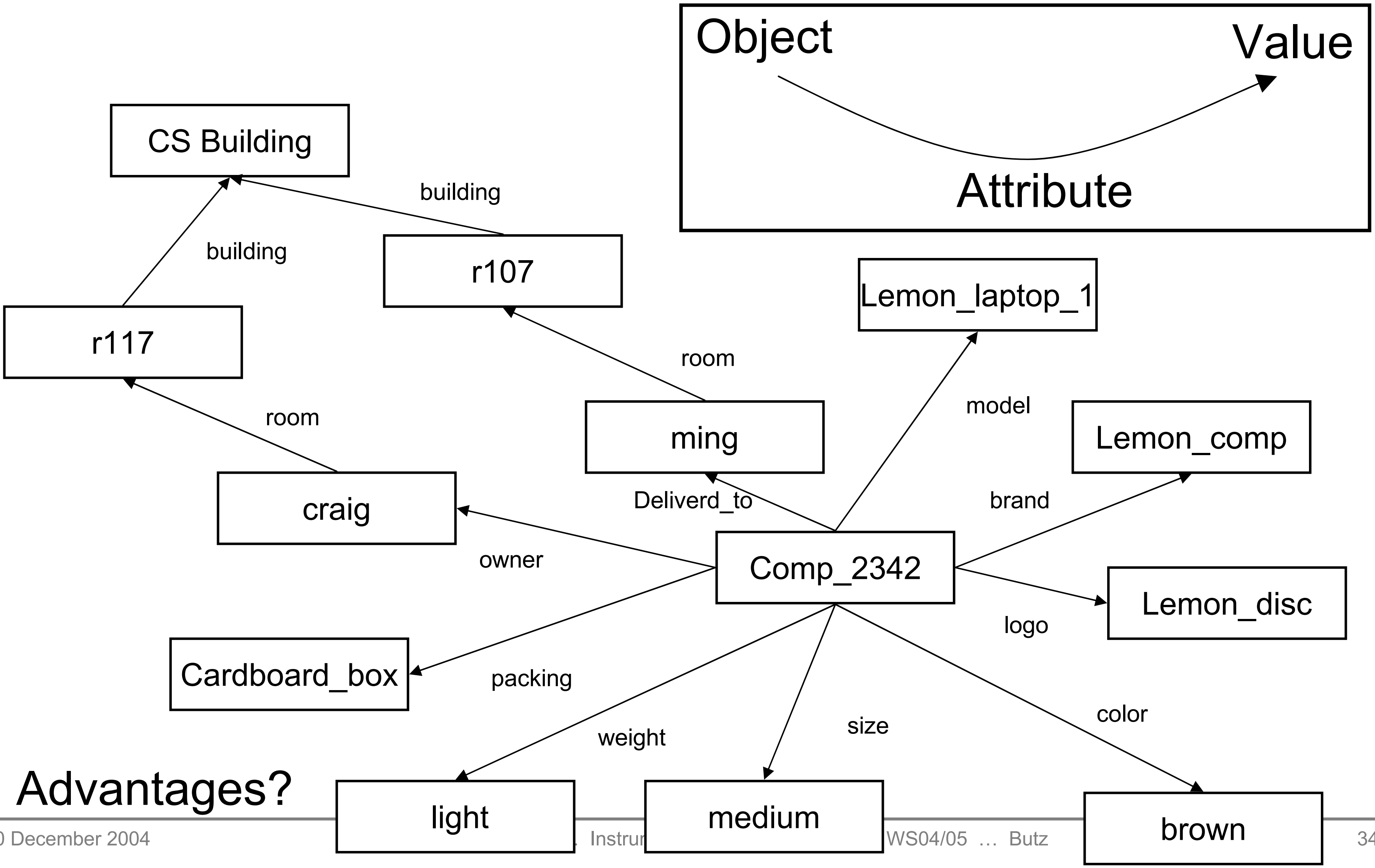
course=c121
time=9:30
room=121

Frame scheduled

instructor=robin
course=c121
time=9:30
room=121
section=2

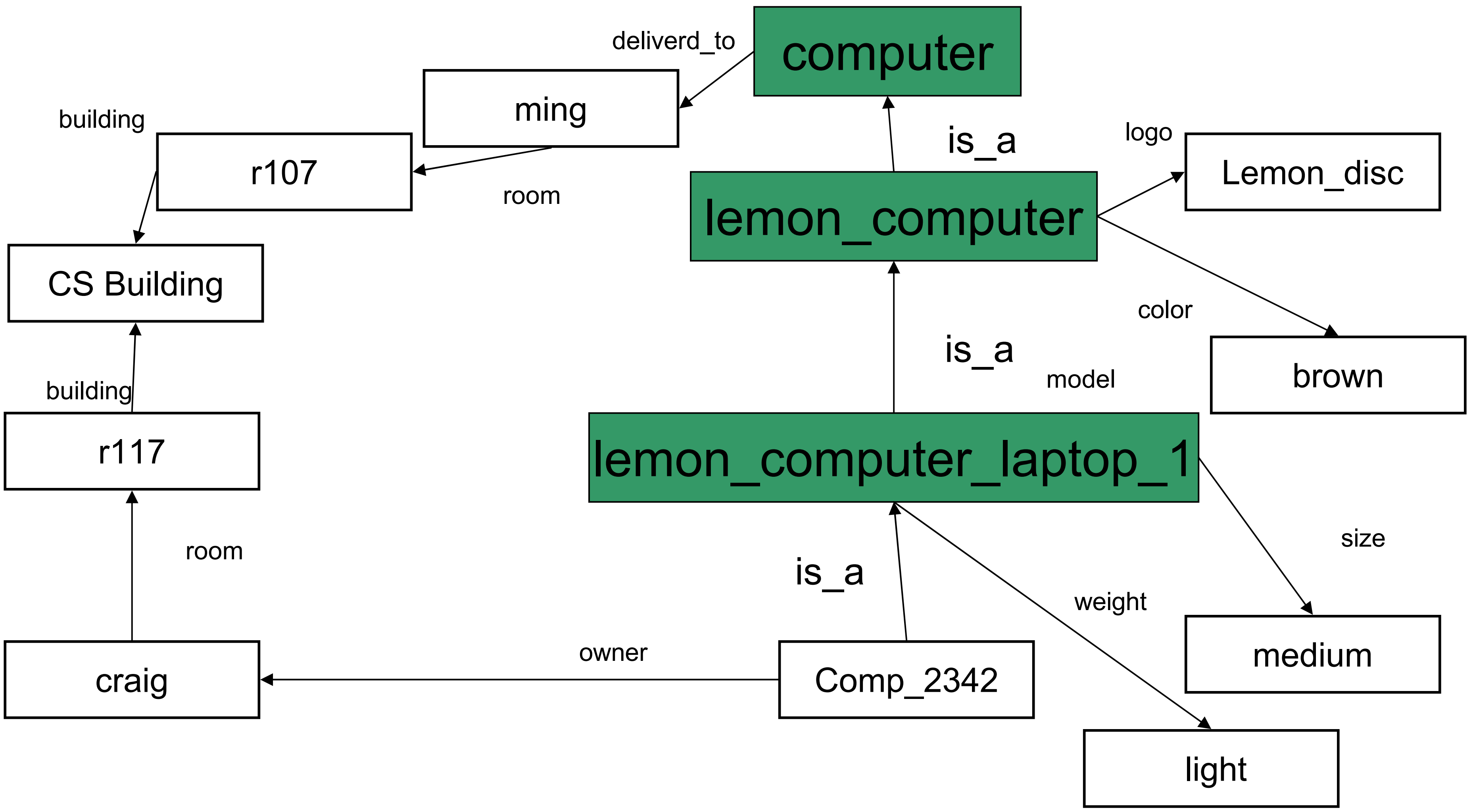
Also works
with variables!

Semantic Networks

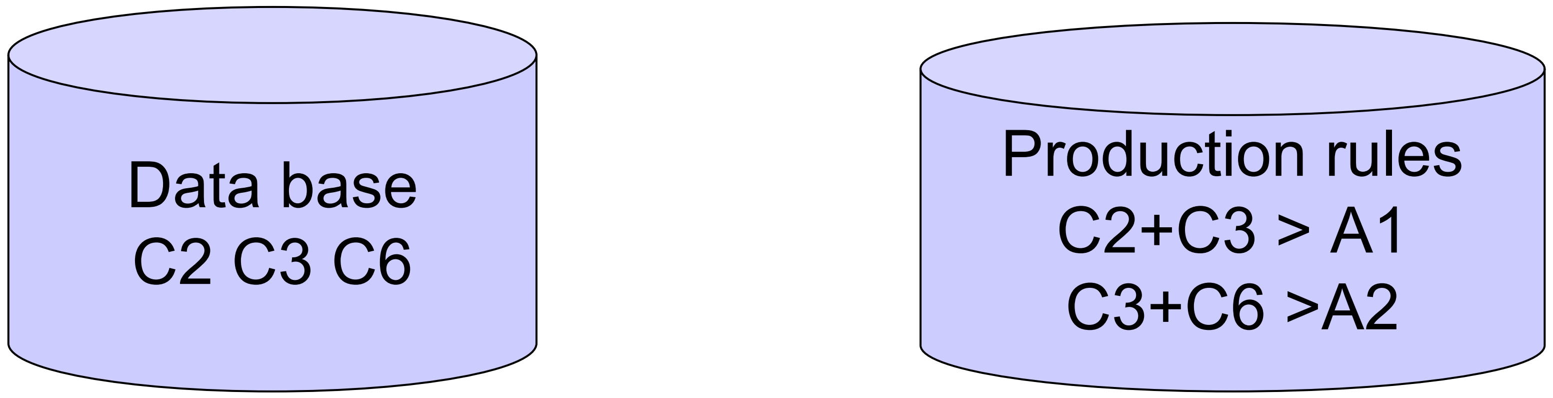


Advantages?

T-Box and A-Box



Production systems



Rule interpreter

1. Matching of Productions
2. Solving the Conflict set
3. Instantiation of a Production
4. Changing the Data base

KR in instrumented environments

- Distributed and dynamic
- Time concept
- Ontologies have to be unified when needed
- Separation of domain dependent and domain independent knowledge
  - E.g.: „knowledge how to explain routes“ vs. „knowledge what the environment looks like“.
- Plan libraries for recognition of user plans

Example: representation of a room

- Motivation: be able to represent/derive the following
  - in_room(E46,Andreas)
  - in_building(Ther. 39,Andreas)
  - left_of(projector, Andreas)
  - prop(playing_darts, Andreas, true)
- How needs the room to be represented in order to derive this?
  - Purely qualitatively?
  - Purely quantitatively?
  - Both? Mixture? Combination?

The role of time

- Reasoning in time
  - The environment changes
  - True statements become false, false statements become true
- Reasoning about time
  - When do certain things happen?
  - In which sequence do they happen?
  - How long do they last?

Representing time

- Different kinds of temporal representations
  - Discrete time: moments
  - Continuous time: intervals
  - Event-based time descriptions
  - Representing time implicitly by state transitions

Logical representations of time

- Actions and time are external to logic
- Integrate time into the logic (Reification)
  - Additional temporal parameter: teaches(Andreas, IE04, E46, 10:15)
- Temporal representation at the meta level
  - Additional predicate: e.g., holds(R,t)
  - holds(teaches(Andreas, IE04, E46), 10:15)
  - Advantage: quantification over relations possible: e.g., $\forall$r: holds(r,t)

Validity / truth

- Representation as a discrete point in time
- Representation as a temporal interval
- Deriving interval properties from point properties
  - Point properties of intervals
    - Valid for all points in I, e.g.,: Robot delivers a package
  - Gestalt properties of intervals
    - Valid only for I as a whole, e.g.,: Robot fulfills his daily duties

Relations in time

- Static relations
  - Validity is not time dependent
- Dynamic relations
  - Validity is time dependent
  - Here also: primitive and derived relations

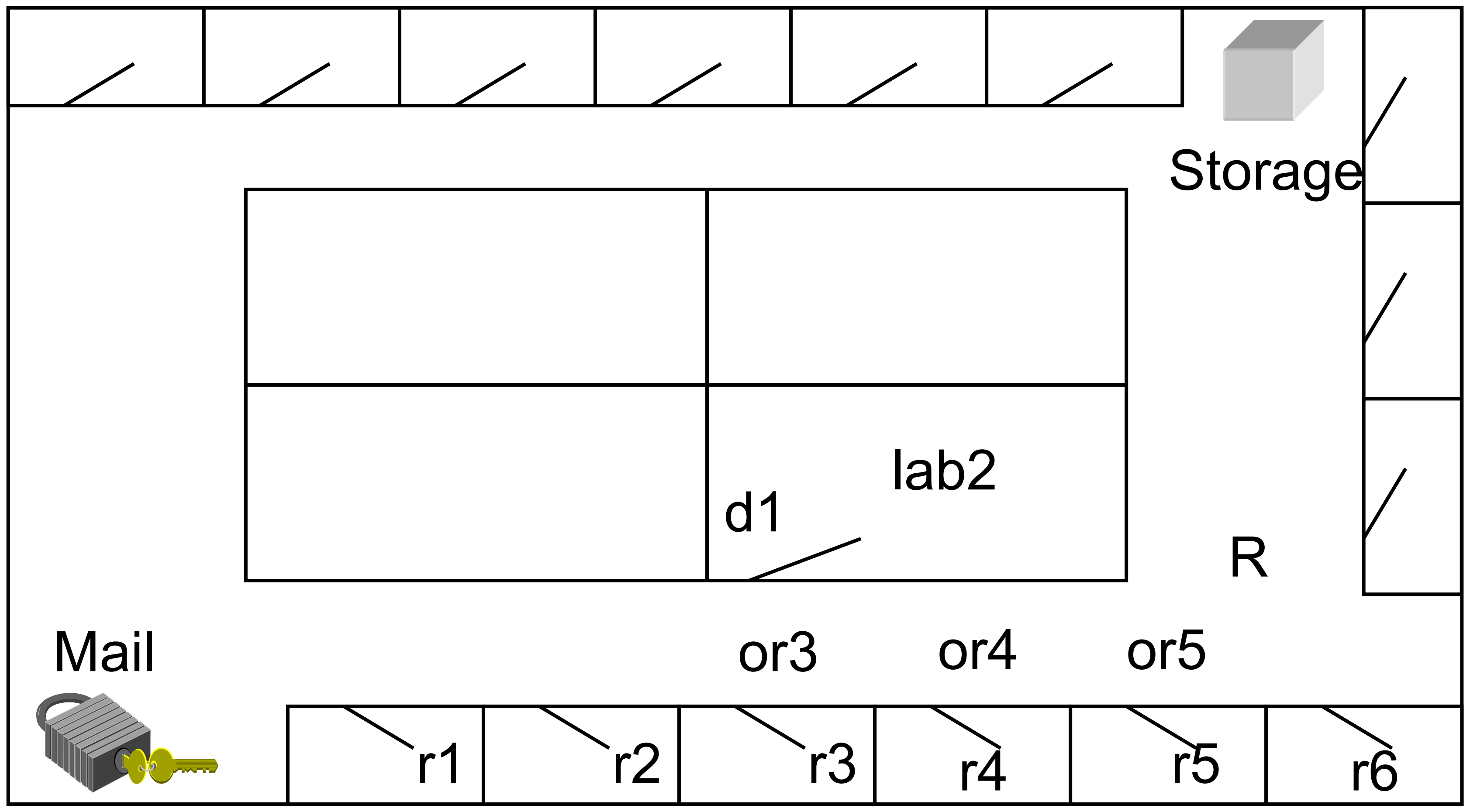
Example relations

- at(Obj, Loc)
- carrying(Ag,Obj)
- sitting_at(Obj,Loc)
- unlocked(Door)
- opens(Key,Door)
- adjacent(Pos1,Pos2)
- between(Door,Pos1,Pos2)

Example actions

- move(Ag,From,To)
- pickup(Ag,Obj)
- putdown(Ag,Obj)
- unlock(Ag,Door)

Example world



Initial state description

- sitting_at(R,or5)
- sitting_at(parcel,storage)
- sitting_at(key,mail)
- between(door1,or3,lab2)
- opens(key,door1)
- adjacent(or1,or2), adjacent(or2,or3),adjacent(or3,or4)......
- adjacent(P1,P2) $\leftarrow$ between(Door,P1,P2) $\wedge$ unlocked(Door).

4 ways of representing temporal sequences of actions

- STRIPS (state based)
- Situation calculus (reified states)
- Event calculus (explicit representation of time)
- Allen‘s interval algebra

STRIPS Stanford Research Institute Problem Solver (1972)

- Specifies the following problem set:
  - Given a state and an action:
    - Can the action be executed?
    - What propositions are still valid after the action?
- STRIPS assumption: there are no further side effects
- STRIPS-representation has 3 parts:
  - Preconditions e.g. [at(Ag,Pos) $\wedge$ sitting_at(Obj,Pos)]
  - Delete List e.g. [sitting_at(Obj, Pos)]
  - Add list e.g. [carrying(Ag,Obj)]

Simplified STRIPS

- Relation poss(A) is true, when action A is possible.
- Example:
  - poss(mov(Ag,Pos1,Pos2) $\leftarrow$ adjacent(Pos1,Pos2) $\wedge$ sitting_at(Ag,Pos1).
  - Eqivalent to the Precondition List in STRIPS

Situation Calculus

- Representation based on situations and actions
- Situations and actions are both reified
- Two major kinds of state
  - init
  - do(A,S)
- Example: do(move(R,or5,or3),init)
- do(A,S) is a partial function: you can‘t do everything in every situation

Representation of predicates

- Meta-level: holds(R,S)
- Add. Argument in the predicate:
  - at(R, or5, init)
  - at(R, or3, do(move(R,or5,or3),init))
  - at(K, mail, do(move(R,or5,or3),init))
- Derived relations can contain situation variables:
  - adjacent(P1,P2,S) $\leftarrow$ between(Door,P1,P2) $\wedge$ unlocked(Door,S).

Preconditions with predicate „poss“

- poss(putdown(Ag,Obj,S)) $\leftarrow$ carrying(Ag,Obj,S).
- Unlocked(Door,do(unlock(Ag,Door),S)) $\leftarrow$ poss(unlock(Ag,Door),S).
- Frame axioms specify what remains unchanged:
  - unlocked(Door,do(A,S)) $\leftarrow$ unlocked(Door,S) $\wedge$ poss(A,S) $\wedge$ A $\neq$ lock(Ag,Door).
  - Carrying(Ag,Obj,do(A,S)) $\leftarrow$ carrying(Ag,Obj,S) $\wedge$ poss(A,S) $\wedge$ A $\neq$ putdown(Ag,Obj).
- „Frame“ here is different from „Frame“ earlier ;-)
  - „Frame Problem“ example: Does a leaf in africa fall from the tree while unlock(Ag,Door) ?

Event Calculus

- Changes in validity are modeled by events at given times
  - Event E happens at time T: event(E,T)
- Specify Consequences of E
  - initiates(E,P,T) : E at time T makes P valid
  - terminates(E,P,T): E at time T makes P invalid

Holds Predicate in Event Calculus

- $holds(P,T) \leftarrow event(E,T0) \wedge T0<T \wedge initiates(E,P,T0) \wedge \neg clipped(P,T,T0).$

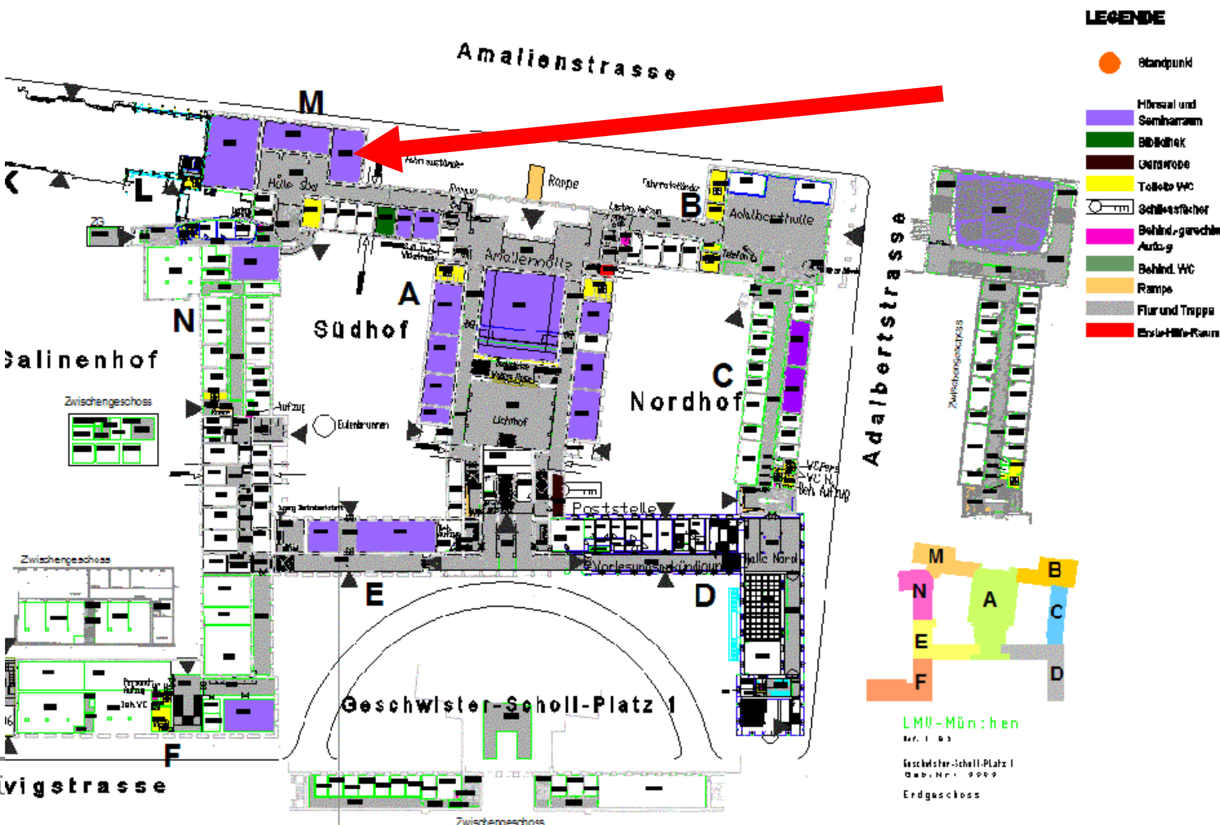
- $clipped(P,T,T0) \leftarrow event(E,T1) \wedge terminates(E,P,T1) \wedge T0<T1 \wedge T1<T.$

Axiomatisation of actions in E.C.

- example:
  - initiates(pickup(Ag,Obj),carrying(Ag,Obj),T) $\leftarrow$ poss(pickup(Ag,Obj),T).
  - terminates(pickup(Ag,Obj), sitting_at(Obj,Pos),T) $\leftarrow$ poss(pickup(Ag,Obj),T).
  - poss(pickup(Ag,Obj),T) $\leftarrow$ Ag $\neq$Obj $\wedge$ holds(at(Ag,Pos),T) $\wedge$ holds(sitting_at(Obj,Pos),T).

Tomorrow: lecture by Marc Böhlen

- **Machines for Supermodernity**
- Dienstag 14.12.04, 12:15 Uhr
  LMU Hauptgebäude, Raum 129 / M010
- Abstract at
  www.mimuc.de


Amalienstrasse
M
L
B
A
N
C
D
E
F
Adalbertstrasse
Südhof
Nordhof
Salinenhof
Lichthof
Eulenbrunnen
Aufzug
Zwischengeschoss
Adalberthalle
Amalienhalle
Rampe
Poststelle
Vorlesungsankündigungen
Halle Nord
Geschwister-Scholl-Platz 1
Ludwigstrasse
LEGENDE
Standpunkt
Hörsaal und Seminarraum
Bibliothek
Garderobe
Toilette WC
Schliessfächer
Behind.-gerechter Aufzug
Behind. WC
Rampe
Flur und Treppe
Erste-Hilfe-Raum
LMU-München
Geschwister-Scholl-Platz 1
Erdgeschoss