Second International Workshop on Agent-Oriented Software Engineering (AOSE'2001), Montreal, Canada, May 29, 2001

Expectation-Oriented Analysis and Design

W. Brauer[*], M. Nickles[*], M. Rovatsos[*], G. Weiß[*], K. F. Lorentzen[+]

[*]Department of Informatics, Technical University of Munich

[+]Department of Technology Assessment, Technical University Hamburg

`http://wwwbrauer.informatik.tu-muenchen.de/gruppen/kikog/projects/socionics/`

Facing the autonomy dilemma

- *Agent autonomy* is a key feature of powerful agent-based software
- Especially in truly *open multiagent systems* (e.g. virtual markets) agent autonomy is crucial and inevitable
- Consequences:
  - The system can provide high flexibility, scalability and robustness
  - Agent behavior cannot fully be predicted and controlled
  - Fully normative system design is neither possible nor desirable
- Engineering methods have to cope with two contrary demands:
  - Preserve agent autonomy and system openness
  - Impose and ensure design specifications

Tackling the autonomy dilemma: The *EXPAND* method

- ***EXPAND*** = *Exp*ectation-Oriented *An*alysis and *D*esign
- Grounding in Niklas Luhmann's *Theory of Social Systems*
- Usage of *expectations* as primary modeling abstraction
- Perspective is on supra-individual social expectations, allowing the designer to design and analyze the *system-level* of the MAS directly
- Knowledge about expectation-grounded social structures is made explicit to the designer as well as to the agents
- Agents are influenced only by means of providing this knowledge, not through exertion of control

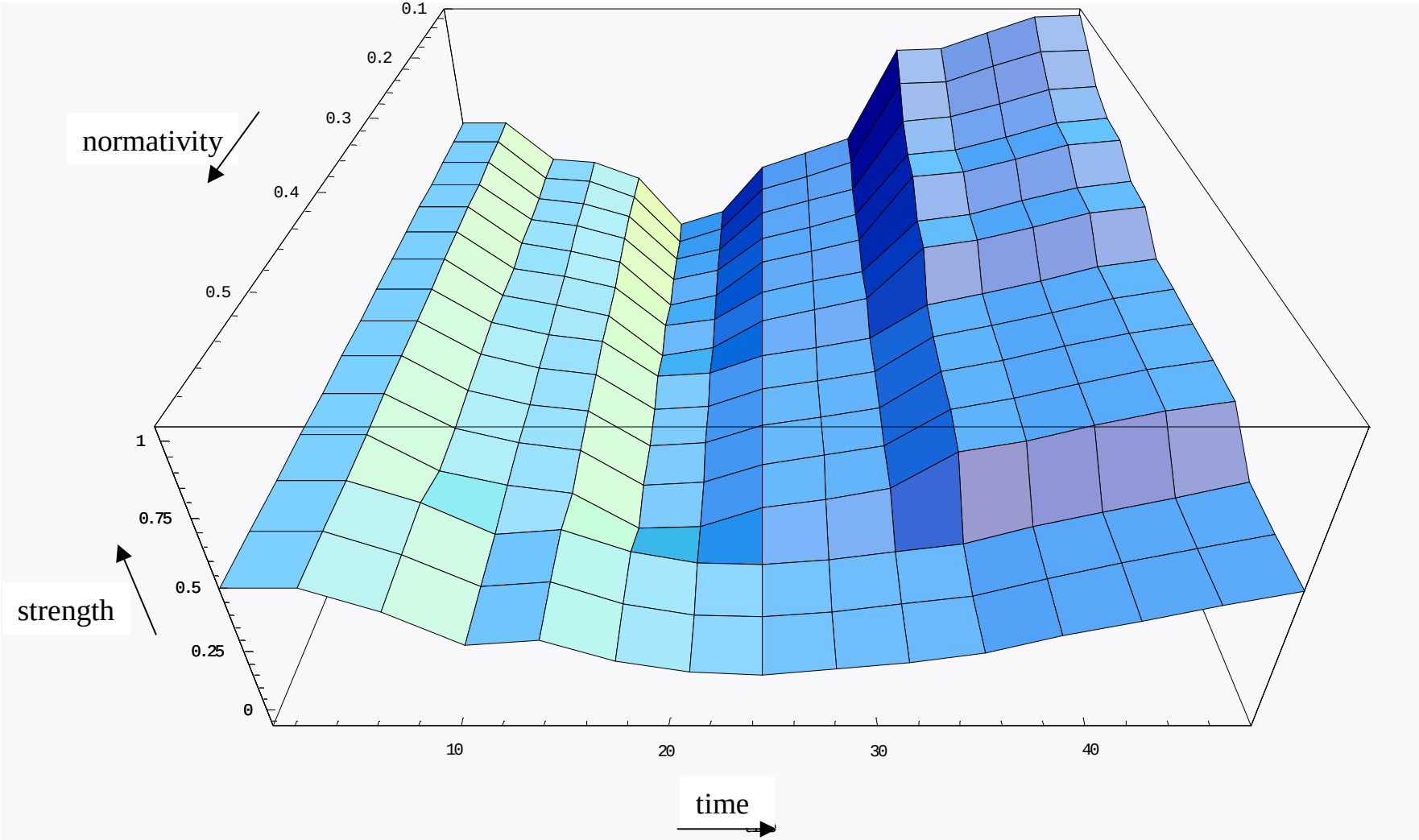
Communications and expectations

- Autonomous agents appear to each other as „black boxes“:
  They cannot fully predict or control the respective other agent
- To overcome this situation, agents need to *communicate*, inducing *sociality*
- Communications are (only) observable as courses of interaction
  $\Rightarrow$ social structures can be modeled by means of *action expectations*
- *System-level expectations* are formed by observers with a global view of the system
- System-level expectations can be derived via statistical evaluation of monitored communication processes

Expectation properties and expectation adaptation

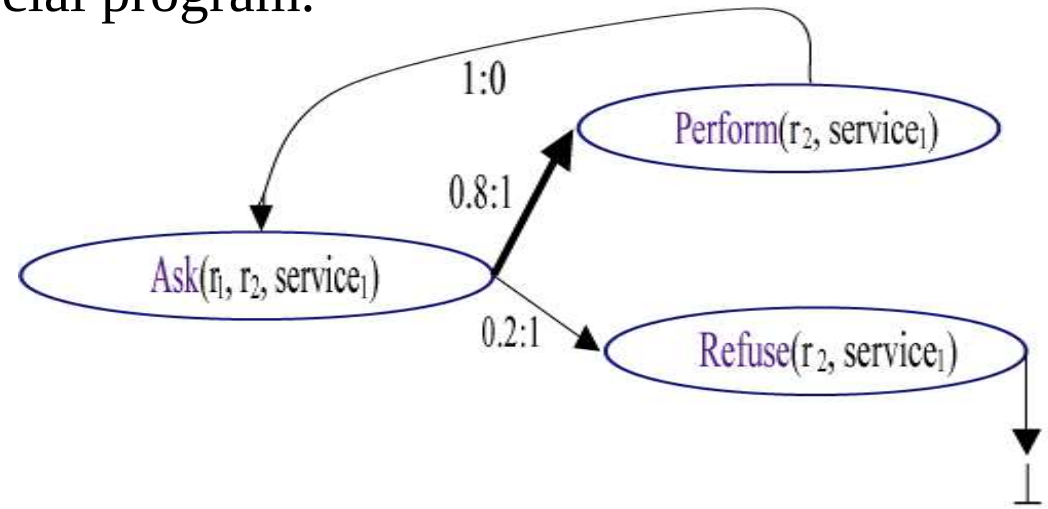
- Key attributes of expectations:
  - ***Strength***: Degree of expectedness. Continually adapted in dependency of the observed agent behavior
  - ***Normativity***: Degree of strength changeability in case of expectation contradiction. Derived and adapted by means of sociological heuristics
  - ***Deviancy***: Difference between expected and actual behavior
- Non-normative (*adaptable*) expectations reflect the probability of the expected behavior
- Normative expectations may be inconsistent with reality (i.e., the actual behavior)

Expectation adaptation (example)



Expectation structures

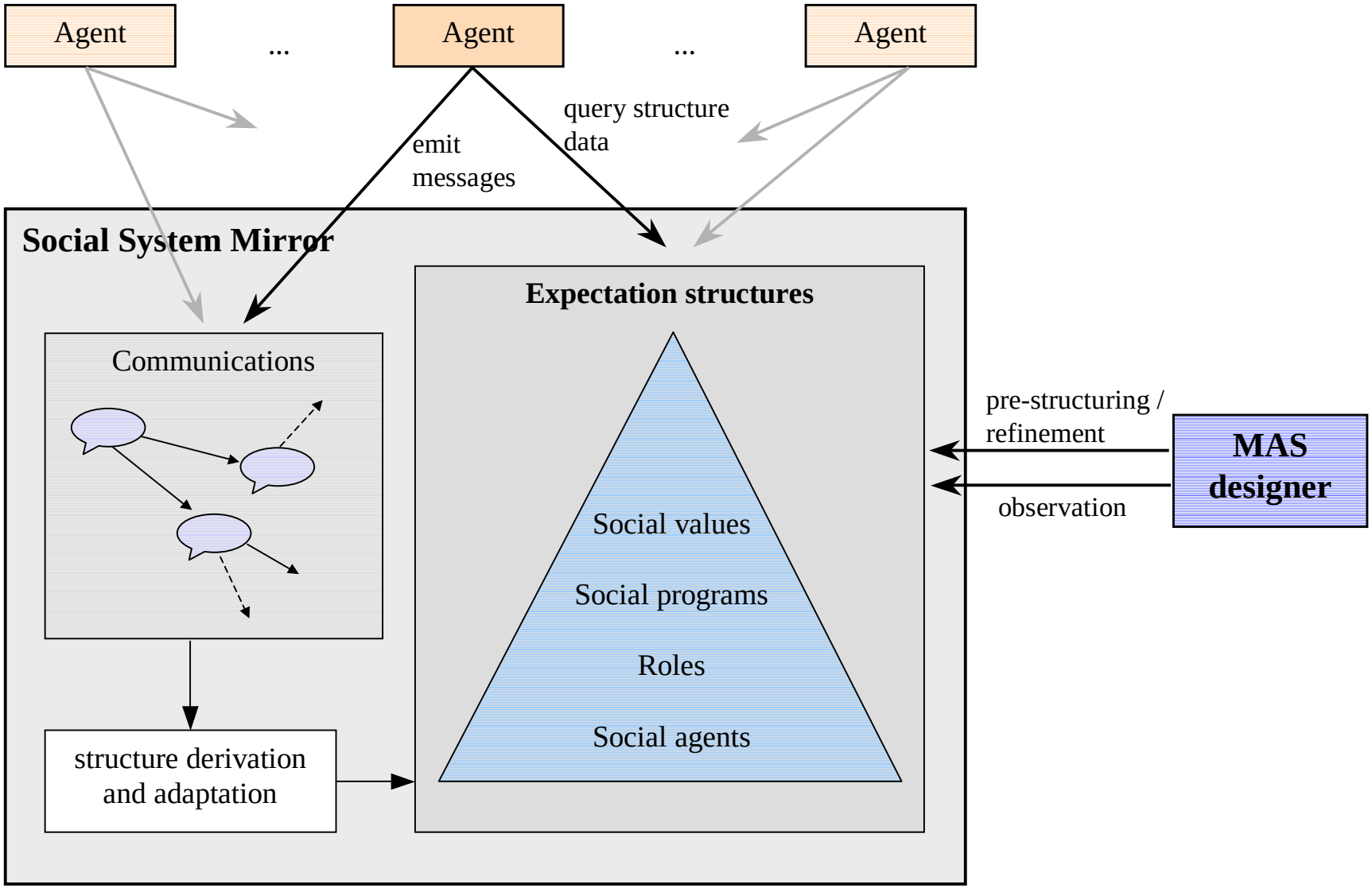
- Expectation structure typology:
  - ***Social agent***: Set of expectations addressed to a single agent
  - ***Role***: Set of expectations addressed to a variable agent representative
  - ***Social program***: Interaction scheme for multiple agents and/or roles
  - ***Social value***: Abstract behavior rating
- A simple social program:



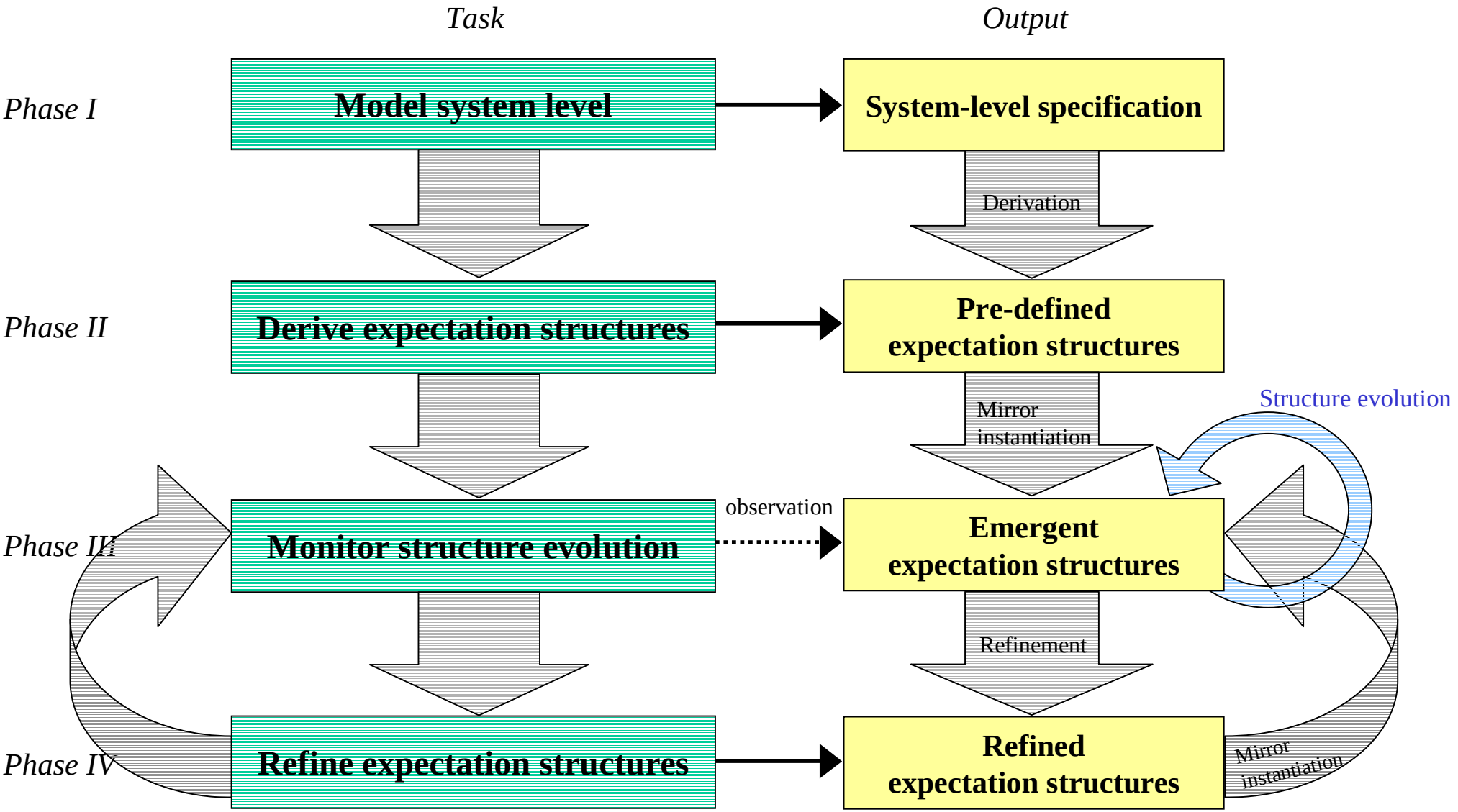
The *Social System Mirror*

- Software component which models the social system of a MAS
- Corresponds to a *CASE tool* for EXPAND
- Purposes:
  - Continual monitoring of agents communication processes
  - Derivation of system-level expectation structures from these observations
  - Making system-level expectation structures visible for the agents (***reflection*** effect) and the designer
- In addition to emergent structures, the mirror also reflects manually designed expectation structures (*pre-structuring*)
- Influence on agents is solely by means of reflection (agents adopt to structures which seem to be useful to them)

The *Social System Mirror* (overview)



The *EXPAND* engineering phases (overview)



Case study: Designing a car trading platform

- Web forum for online car trading
- Buyers and sellers are represented by agents
- Open system, autonomous „black box“ trading agents
- Design goals:
  - Broad acceptance of the platform, high site traffic
  - High reliability of transactions
  - Reasonable trust between buyers and sellers
  - Maximum transaction turnover

Engineering phase I: Modeling the system level

- Specification of the desired *social behavior* and *social functionality* in terms of ...
  - Goals
  - Social plans
  - Constraints, rules
  - Desired environment states
  - System-level expectation structures
    (social agents, roles, social programs, social values)

  ...

Case study: *Engineering phase I*

- Modeling the system level results in the following specifications:

(1) Agents must fulfill obligations (pay/deliver) resulting from committing themselves to purchasing/selling cars

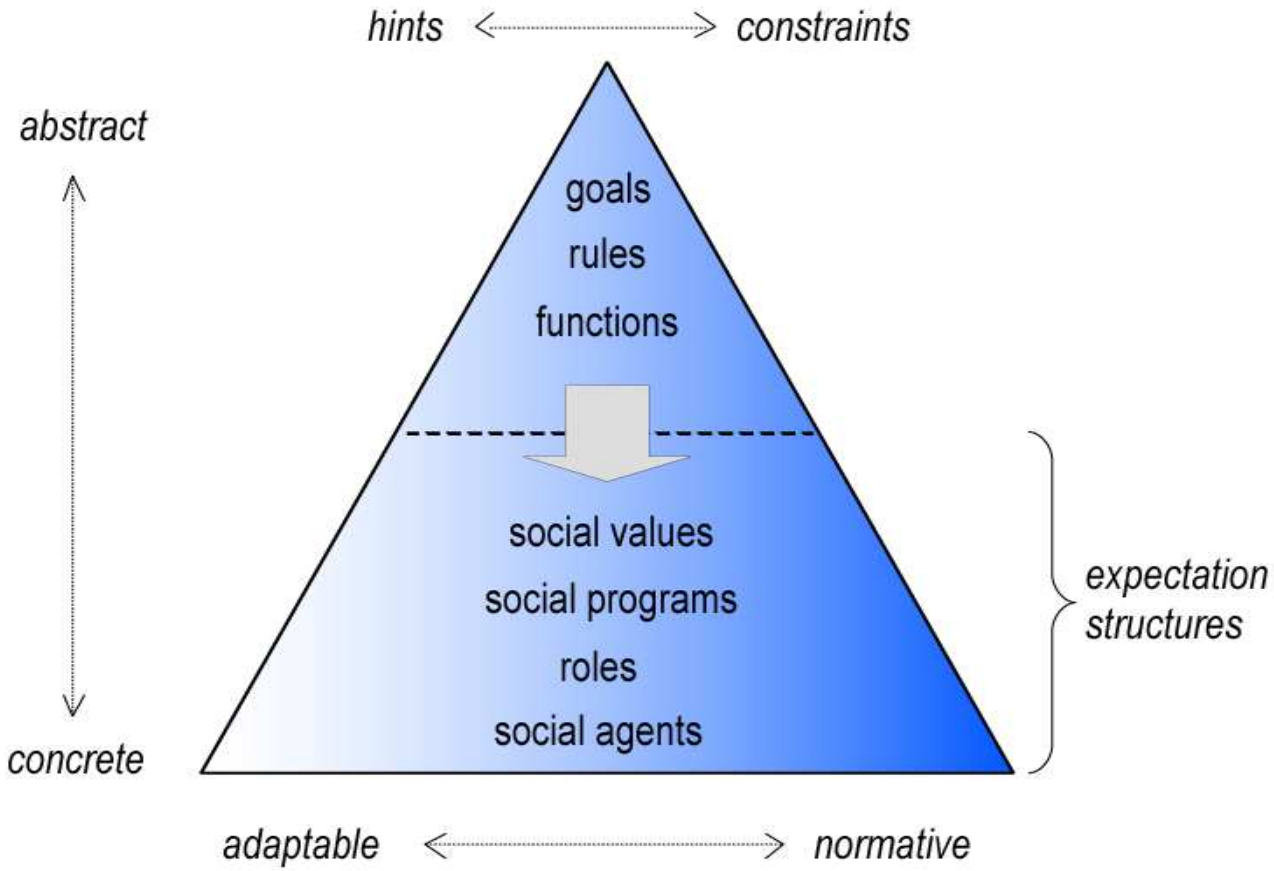
(2) Unreliable behavior induces reluctance to enter business relationships

(3) Fraudulence leads to exclusion from the trading platform

(4) To keep up the use of the platform, interest in offers and requests should be communicated

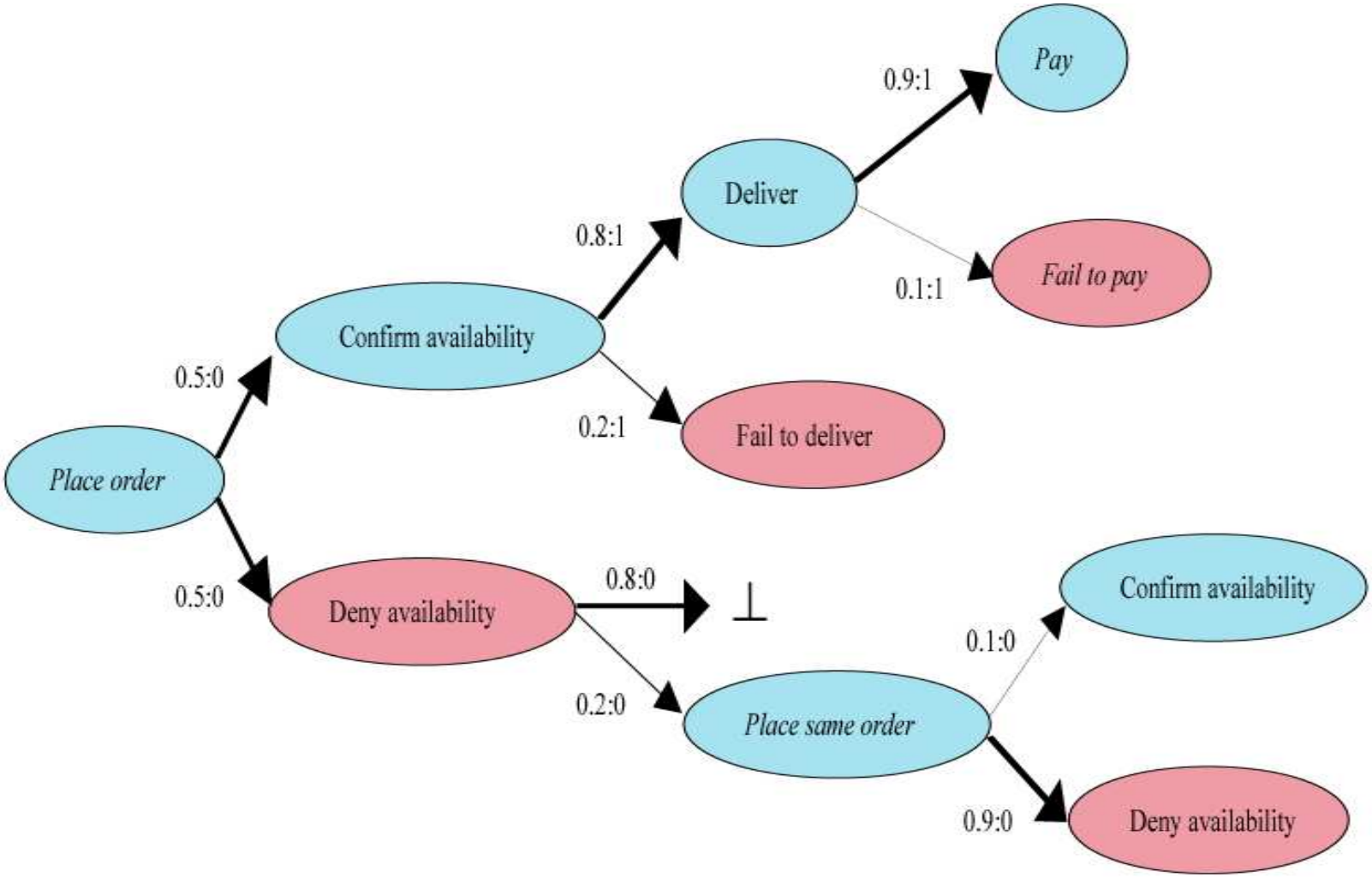
Engineering phase II: Deriving expectation structures

- Derivation of system-level expectation structures from the design specifications of phase I
- Announcement via Social System Mirror



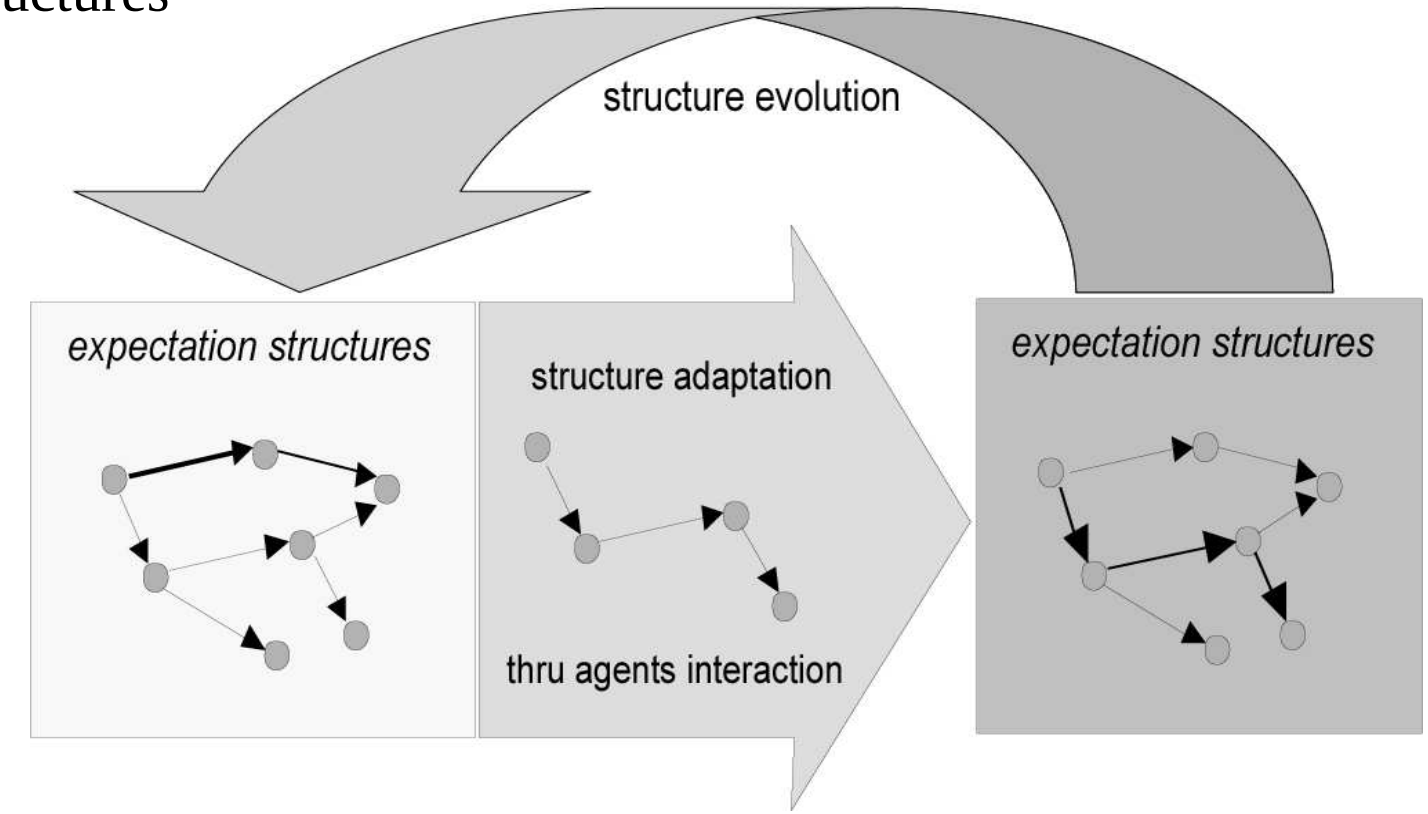
Case study: *Engineering phase II*

- Deriving expectation structures (here derived from specification (1))



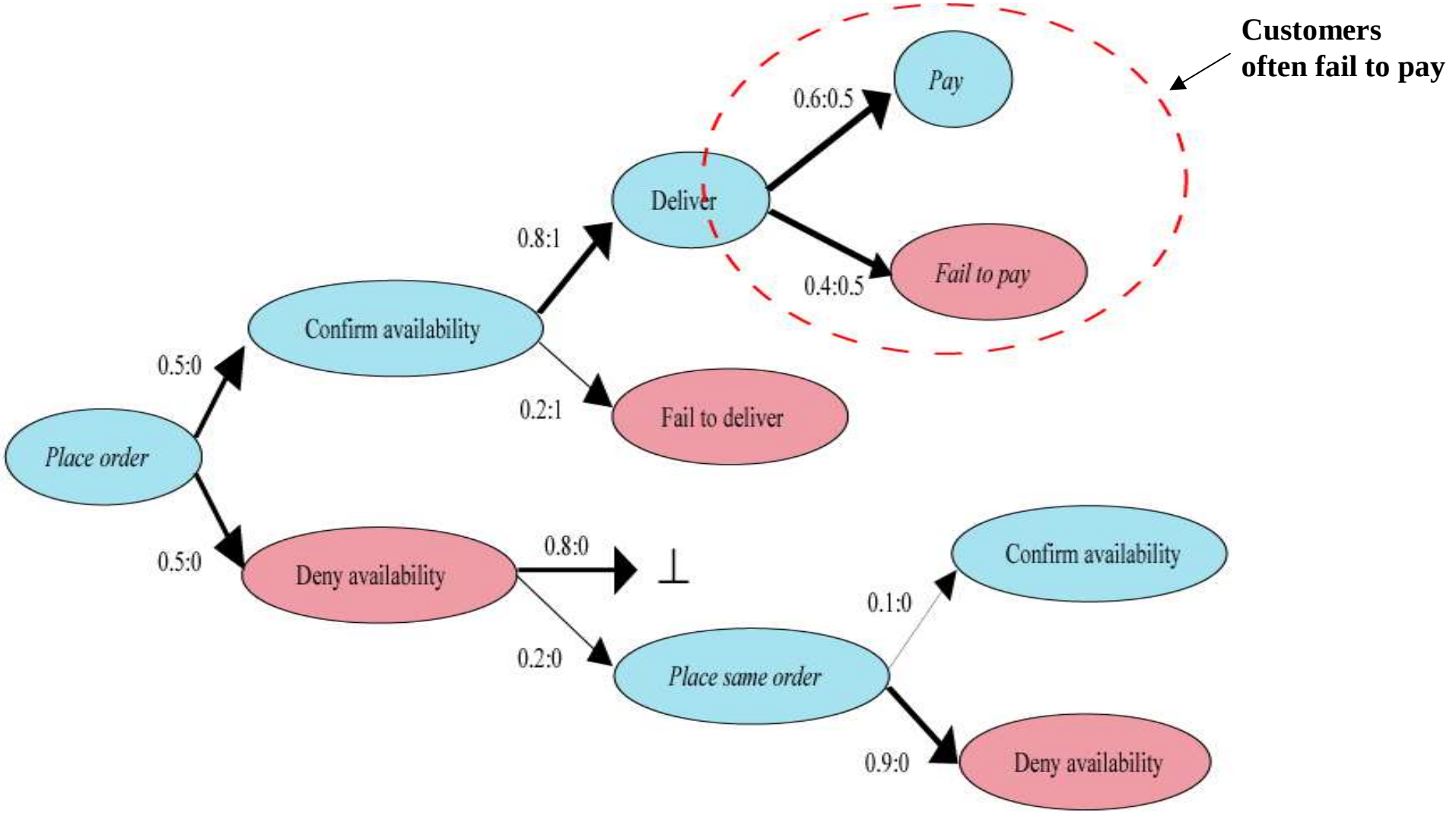
Engineering phase III: Monitoring structure evolution

- Observation of the *expectation structure evolution* shown by the Social System Mirror
- Identification of emergent „positive“ and „negative“ expectation structures



Case study: *Engineering phase III*

- Observation and identification of interesting emergent structures

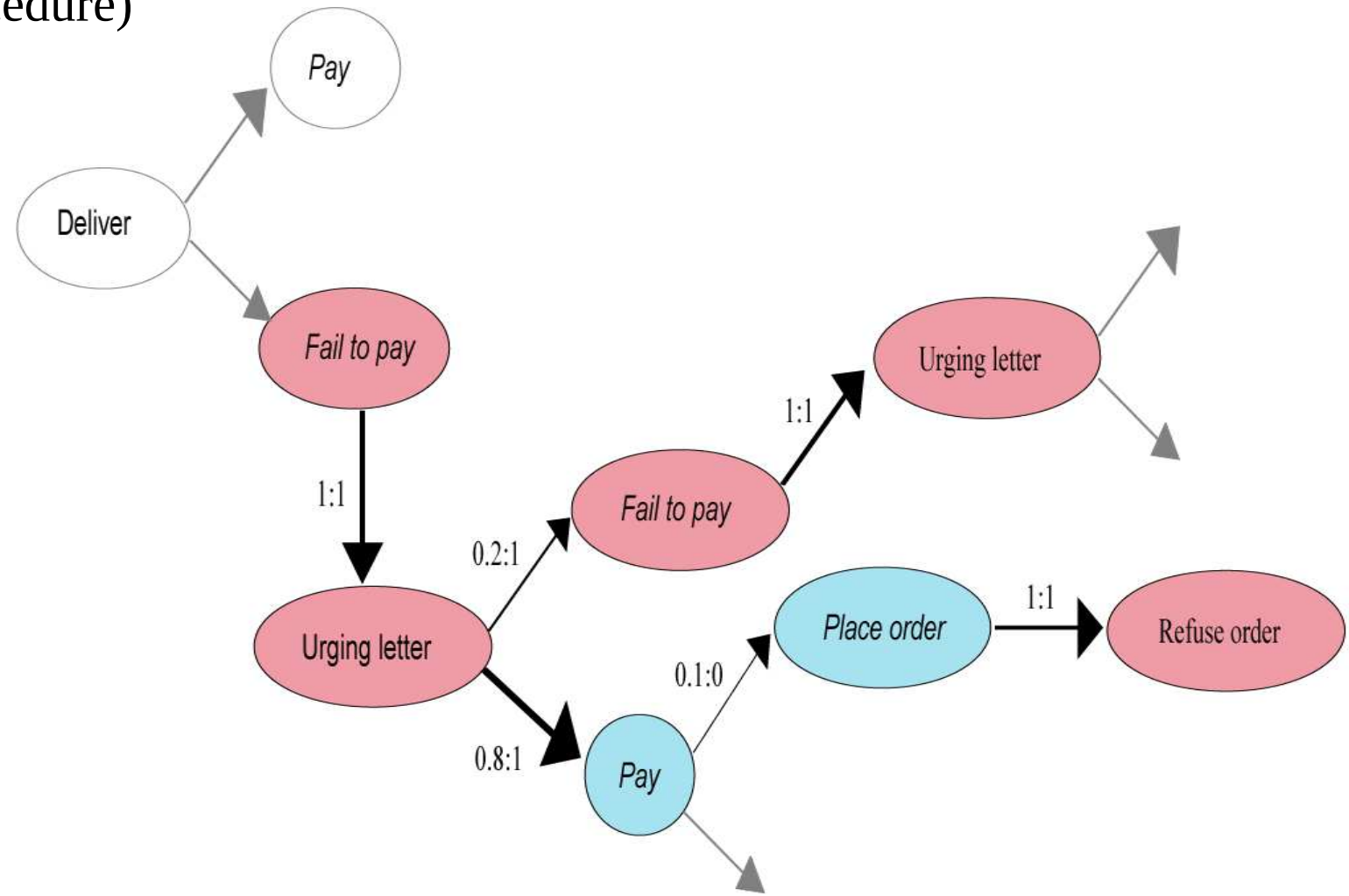


Engineering phase IV: Refining expectation structures

- Refinement of expectation structures within the Social System Mirror, depending on the outcome of phase III:
  - Removal of negative emergent expectation structures
  - Strengthening of useful expectation structures
  - Adding of new useful expectation structures
- Proceeding to phase III until design goals are achieved or no further improvement is expectable

Case study: *Engineering phase IV*

- Refinement of expectation structures (here: adding of a reminder procedure)



Conclusions

- EXPAND is related to engineering methods which focus on the analysis and design of the *system level* (e.g. *Gaia*), but has a fundamentally different concept:
  - EXPAND admits agents a maximum degree of autonomy, preserving the difference between ordinary software objects and agents
  - EXPAND is thoroughly grounded in social theory, enabling a socially oriented perspective on MAS
- Further development is a long-term effort, focusing on
  - Formal treatment of expectation structures
  - Technical refinement of the Social System Mirror
  - Transition from system-level down to the subsystem- and agent-level

Thank you very much for your attention !