

# Project BaBe

(Bayesian Behaviour Networks)

Engineering Emergence  
in a  
Complex Adaptive Enterprise

Dr. Anet Potgieter



**COMPLEX ADAPTIVE SYSTEMS**  
EVOLVING WITH EMERGENCE





# Overview

1. Complex Adaptive Systems
2. Complex Adaptive Enterprises
3. Engineering of Emergence
4. Evolving Emergent Networks





# What is a Complex Adaptive System ?

- **Acquires information** about
  - environment
  - interaction with that environment
- Identify **regularities**
- Condense regularities into a **model**
- **Act in the real world** using the Murray Gell-Mann model





# The Complex Adaptive Enterprise

The **Complex Adaptive Enterprise** is a complex adaptive system.

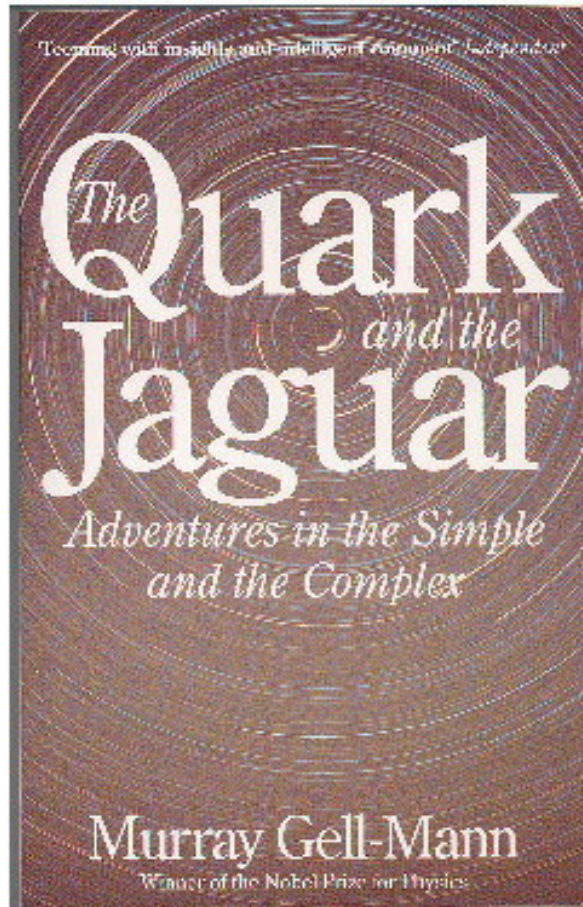
Sustains competitive advantage

- Under conditions of **uncertainty, complexity** and **constant change**
- By **learning from experience**
- By being **self-aware**
- By **adapting its behaviour** in order to constantly outperform its competitors





# Complexity Theory



*"The world of the quark has everything to do with a jaguar circling in the night"*

- Arthur Sze (cited by Murray Gell-Mann)



# What is Emergence?

- The most important characteristic of a Complex Adaptive System
- The **collective behaviour** of interacting system components



## Emergence Test: Design, Observation, Surprise!

Ronald, Sipper & Capcarrère defined the *emergence test* in terms of three conditions, namely

- Design
- Observation
- Surprise

of a system designer and a system observer  
(which could be the same)



# Design

- The system has been constructed by the designer
- *local interactions* between components described in a language  $\mathcal{L}_1$ .



## Observation

- The observer is *fully aware* of the design
- Describes *global behaviours* and properties of the running system, over a period of time,
- Uses a language  $\mathcal{L}_2$ .

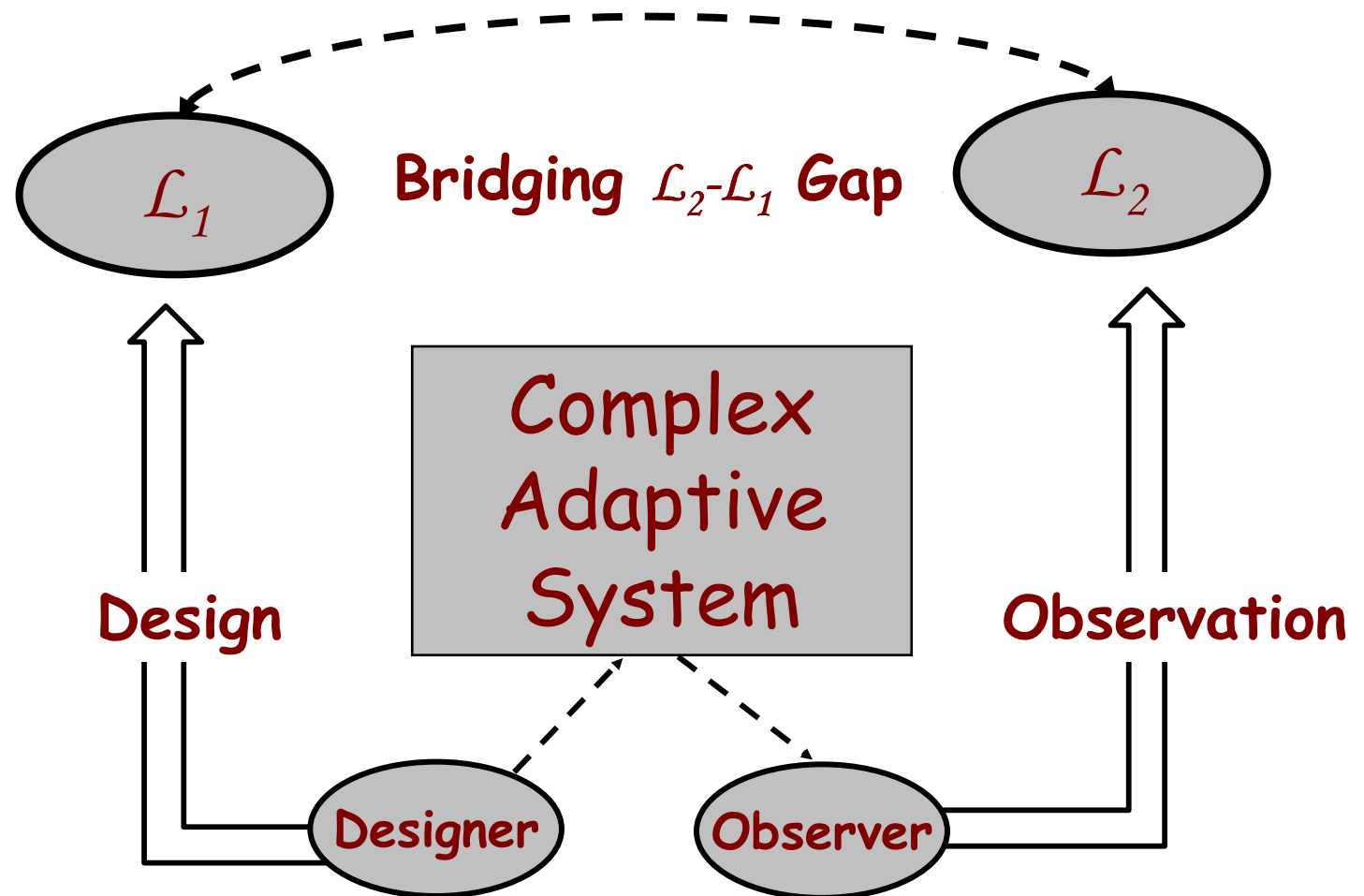


## Surprise !

- The language of design  $\mathcal{L}_1$  and the language of observation  $\mathcal{L}_2$  are distinct
- **Causal link** between the elementary interactions programmed in  $\mathcal{L}_1$  and the behaviours observed in  $\mathcal{L}_2$  is ***non-obvious*** to the observer
- Observer experiences ***surprise***.



# Emergent Engineering





# The Engineering of Emergence

- *Shuffling* back and forth between  $\mathcal{L}_1$  and  $\mathcal{L}_2$
- *Changing* things on the one side
- *Checking* the effects on the other side



# Classical Engineering

## Intolerance of surprises

### ➤ *Requirement Specifications*

Language of (desired) observation  $\mathcal{L}_2$   
formulated in consultation with client

### ➤ *System and software design* -

Language of design  $\mathcal{L}_1$  derived from  $\mathcal{L}_2$

### ➤ *Implementation and Unit-testing* -

measured against design  $\mathcal{L}_1$

### ➤ *Integration and System Testing* -

measured against observation  $\mathcal{L}_2$ .

**Elimination of all surprises** either by  
**adapting**  $\mathcal{L}_2$  or **adapting**  $\mathcal{L}_1$  and implement  
changes in the system

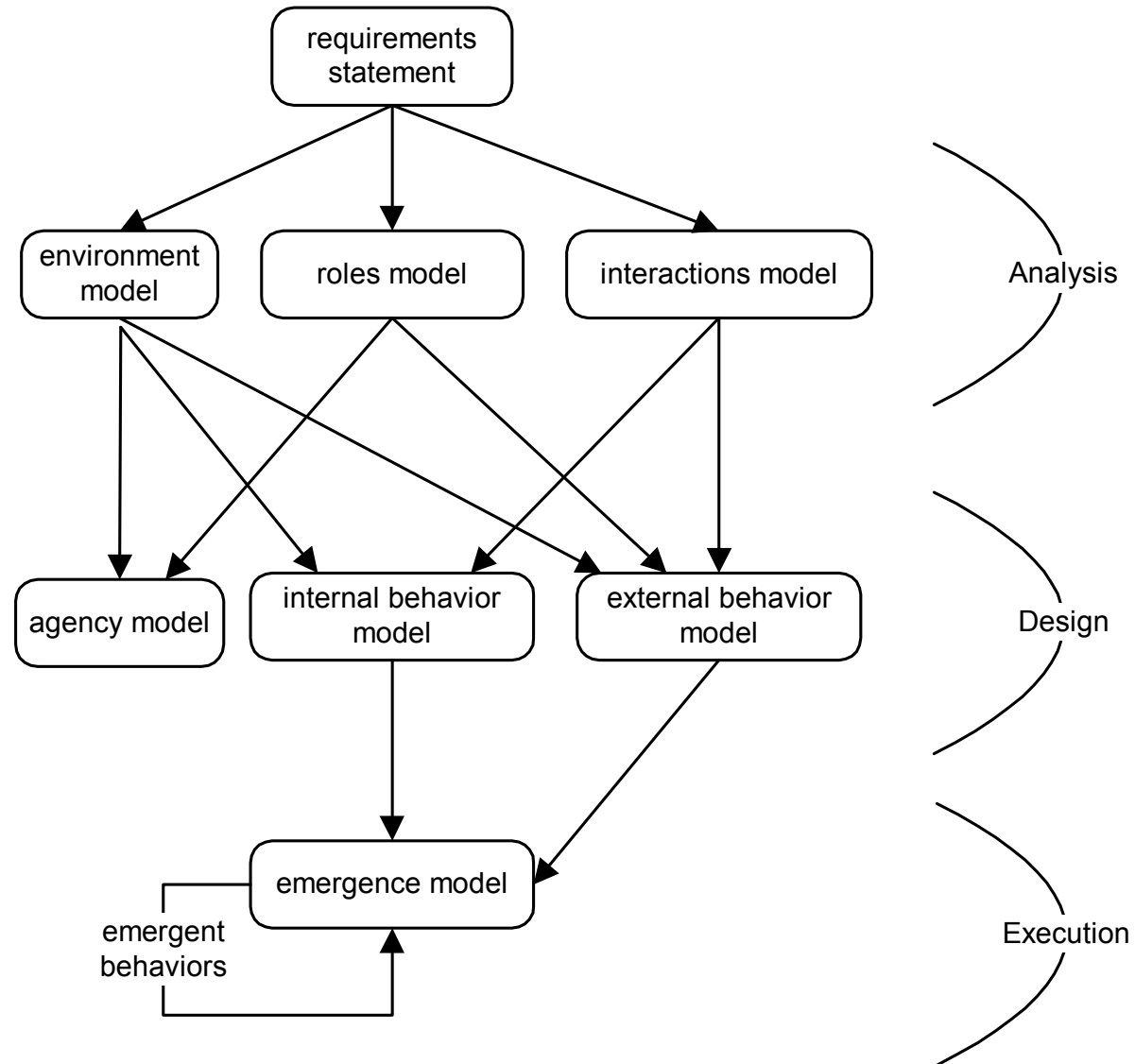


# Emergent Engineering

- *Manages surprises* as part of the engineering lifecycle
- Management of a *persistent*  $\mathcal{L}_1$ - $\mathcal{L}_2$  understanding gap



# AOSE and CAS





# Discussion

