Logics and Calculi for All

Workshop dedicated to L. S. Barbosa on the occasion of his 60th Anniversary

Formal Methods: from the Cathedral of 'Components as Coalgebras' to the Open Source Software Bazar

Antonio Cerone

antonio.cerone@nu.edu.kz

Department of Computer Science, Nazarbayev University, Kazakhstan

Formal Methods: the Beginning

1962

- Carl Adam Petri's PhD thesis
  "Kommunikation mit Automaten"
  (Communication with Automata)

Formal Methods: the Beginning

1962

- Carl Adam Petri's PhD thesis
  "Kommunikation mit Automaten"
  (Communication with Automata)
  $\Longrightarrow$ underspecified title:
  - Does it mean that automata communicate with each other?

Formal Methods: the Beginning

1962

- Carl Adam Petri's PhD thesis
  "Kommunikation mit Automaten"
  (Communication with Automata)
  $\Longrightarrow$ underspecified title:
  - Does it mean that automata communicate with each other?
  - Or does it mean that automata communicate with humans?

Formal Methods: the Beginning

1962

- Carl Adam Petri's PhD thesis
  "Kommunikation mit Automaten"
  (Communication with Automata)
  $\Longrightarrow$ underspecified title:
  - Does it mean that automata communicate with each other?
  - Or does it mean that automata communicate with humans?
  - Or with an environment comprehensive of humans?

Formal Methods: the Beginning

1962

- Carl Adam Petri's PhD thesis
  "Kommunikation mit Automaten"
  (Communication with Automata)
  $\Longrightarrow$ underspecified title:
  - Does it mean that automata communicate with each other?
  - Or does it mean that automata communicate with humans?
  - Or with an environment comprehensive of humans?
  - Or that they allow humans to communicate with each other?

Formal Methods: the Beginning

1962

- Carl Adam Petri's PhD thesis
  "Kommunikation mit Automaten"
  (Communication with Automata)
  $\Longrightarrow$ underspecified title:
  - Does it mean that automata communicate with each other?
  - Or does it mean that automata communicate with humans?
  - Or with an environment comprehensive of humans?
  - Or that they allow humans to communicate with each other?
- Luís Soares Barbosa was born on 4 February

Goal of Formal Methods

1962 — Carl Adam Petri's PhD thesis envisaged a two-fold goal for formal modelling techniques:

1. computational description of the system behaviour

Goal of Formal Methods

1962 — Carl Adam Petri's PhD thesis envisaged a two-fold goal for formal modelling techniques:

1. computational description of the system behaviour, inspired by
   - laws of physics in introducing locality
   - preservation laws present in chemical equations $\Longrightarrow$ reversibility of processes

Goal of Formal Methods

1962 — Carl Adam Petri's PhD thesis envisaged a two-fold goal for formal modelling techniques:

1. computational description of the system behaviour, inspired by
   - laws of physics in introducing locality
   - preservation laws present in chemical equations $\Longrightarrow$ reversibility of processes
2. support a real-life, pragmatic description of the system usage by humans

Proliferation of Formal Methods

- Petri nets
- Process Algebras (PA):
  CCS, CSP, ACP
  - Timed PA
  - Probabilistic PA: PEPA/PRISM
- Model-based Specification Approaches:
  VDN, Z, B
- Rewriting Logic:
  Maude, CafeObj, ELAN

Proliferation of Formal Methods

- Petri nets
- Process Algebras (PA):
  CCS, CSP, ACP
  - Timed PA
  - Probabilistic PA: PEPA/PRISM
- Model-based Specification Approaches:
  VDN, Z, B
- Rewriting Logic:
  Maude, CafeObj, ELAN

Limited in terms of data structure and control flow!

Component as Coalgebras

2001 — Luís Soares Barbosa's PhD thesis:

- duality of system components: static ('data') vs dynamic ('behaviour') structure

Component as Coalgebras

2001 — Luís Soares Barbosa's PhD thesis:

- duality of system components: static ('data') vs dynamic ('behaviour') structure
- $\Longrightarrow$ formal basis: duality between algebraic and coalgebraic structures in computation

Component as Coalgebras

2001 — Luís Soares Barbosa's PhD thesis:

- duality of system components: static ('data') vs dynamic ('behaviour') structure
- $\Longrightarrow$ formal basis: duality between algebraic and coalgebraic structures in computation
- $\Longrightarrow$ 'Components as Coalgebras': semantic framework to which process calculi and popular notations (e.g. UML) can be mapped

Component as Coalgebras

2001 — Luís Soares Barbosa's PhD thesis:

- duality of system components: static ('data') vs dynamic ('behaviour') structure
- $\Longrightarrow$ formal basis: duality between algebraic and coalgebraic structures in computation
- $\Longrightarrow$ 'Components as Coalgebras': semantic framework to which process calculi and popular notations (e.g. UML) can be mapped $\Longrightarrow$ rigorous component development and behavioural refinenment for componentd

Add External Mechanisms

Formal notations may be extended by

- adding mechanism to define complex data types
- introducing more general control flow structures

Approach used in PAT (Process Analysis Toolkit)

Add External Mechanisms

Formal notations may be extended by

- adding mechanism to define complex data types
- introducing more general control flow structures

Approach used in PAT (Process Analysis Toolkit)

But

- implementation-oriented control structures
- richness of available data types

$\Longrightarrow$ negative consequences: forget original logical capabilities and abuse new features

Grow Internally

Formal notations may combine

- functional approach to define data types
- rewrite logic to provide control flow

Approach used in Maude

Grow Internally

Formal notations may combine

- functional approach to define data types
- rewrite logic to provide control flow

Approach used in Maude

Any extension is grown internally using the core language

Formal Methods ...

- may use a large variety of mathematical structures and representation techniques for system states and evolution

Formal Methods ...

- may use a large variety of mathematical structures and representation techniques for system states and evolution
- can be applied to a number of domains, even outside computer science (multidisciplinary)

Formal Methods ...

- may use a large variety of mathematical structures and representation techniques for system states and evolution
- can be applied to a number of domains, even outside computer science (multidisciplinary)
- can be used at the meta-level to provide the semantics of programming languages, other formal methods and higher-level and domain-specific notations

Formal Methods ...

- may use a large variety of mathematical structures and representation techniques for system states and evolution
- can be applied to a number of domains, even outside computer science (multidisciplinary)
- can be used at the meta-level to provide the semantics of programming languages, other formal methods and higher-level and domain-specific notations
- provide a tool capable to combine, armonise and link multidisciplinary concepts under a unified semantics (interdisciplinary)

Interdisciplinarity: FM and HCI

- HCI is an interdisciplinary dicipline
  $\Longrightarrow$ how the two worlds of computer science and human psychology meet each other

Interdisciplinarity: FM and HCI

- HCI is an interdisciplinary dicipline
  $\Longrightarrow$ how the two worlds of computer science and human psychology meet each other
- initially two distinct perspectives, from psychology and from software engineering

Interdisciplinarity: FM and HCI

- HCI is an interdisciplinary dicipline
  $\Longrightarrow$ how the two worlds of computer science and human psychology meet each other
- initially two distinct perspectives, from psychology and from software engineering
- 1980s: mathematical-logical formalisms used to describe not just computer systems, but the system usage by humans

HCI: FM Backward vs Forward

Application to safety-critical systems

- backward perspective considers either the formal description of expected effective behaviour or the formal analysis of errors performed by the operator as reported by accident analysis

HCI: FM Backward vs Forward

Application to safety-critical systems

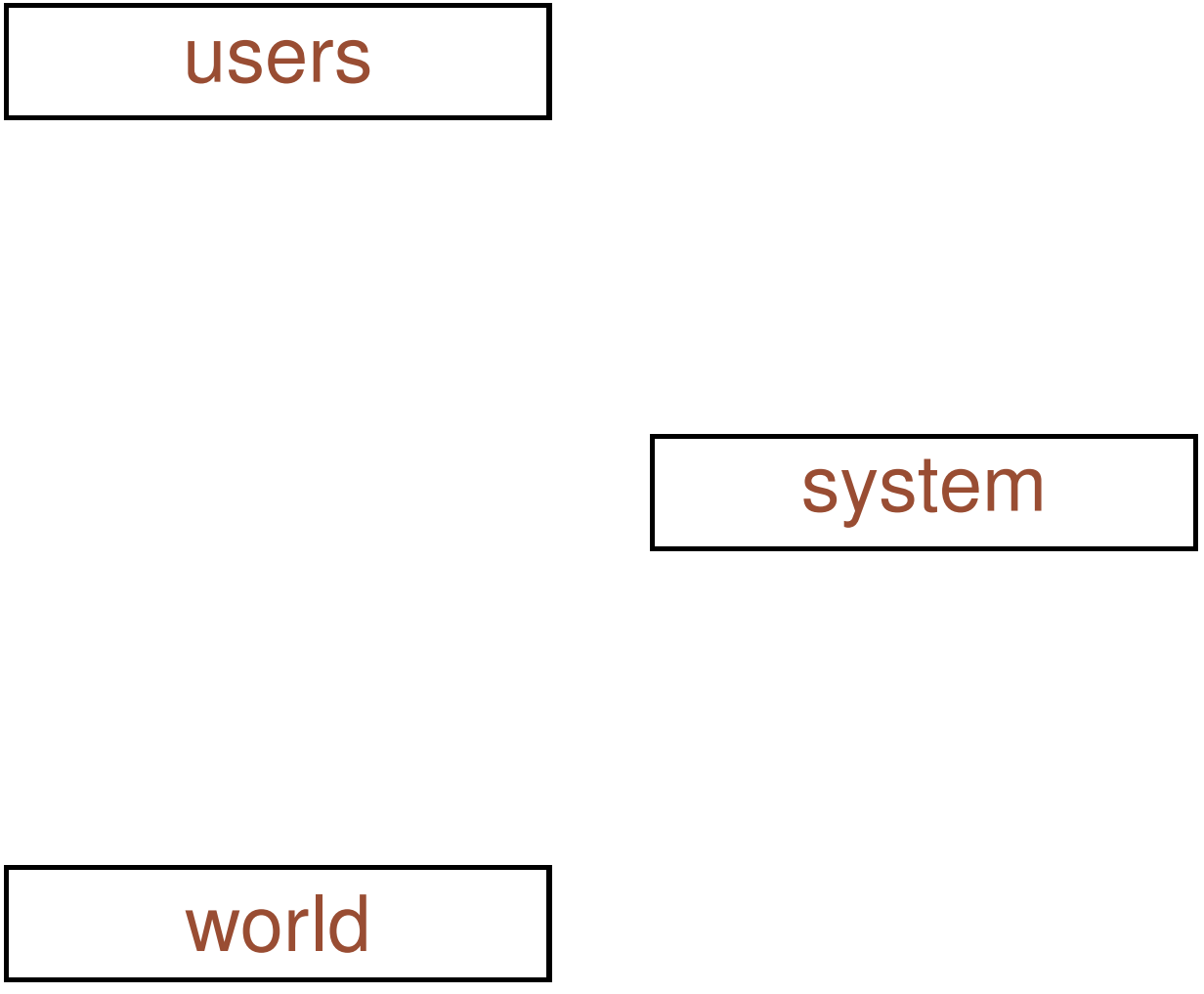
- backward perspective considers either the formal description of expected effective behaviour or the formal analysis of errors performed by the operator as reported by accident analysis
- forward perspective defines a cognitive plausible user behaviour, based on formal assumptions to bind the users act driven by cognitive processes and composes such a cognitive model with the system model

In Terms of Category Theory

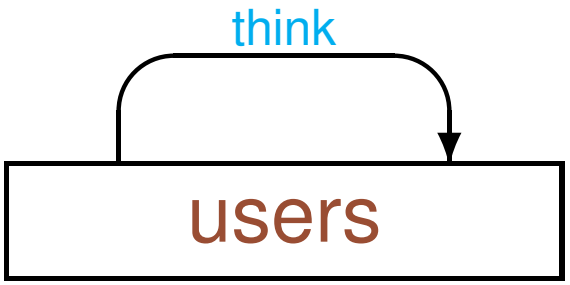
Application to safety-critical systems

- backward perspective
  - backward morphism
  - backward refinement
- forward perspective
  - forward morphism
  - forward refinement
    $\Longrightarrow$ non-determinism reduction

HCI: FM Forward Perspective



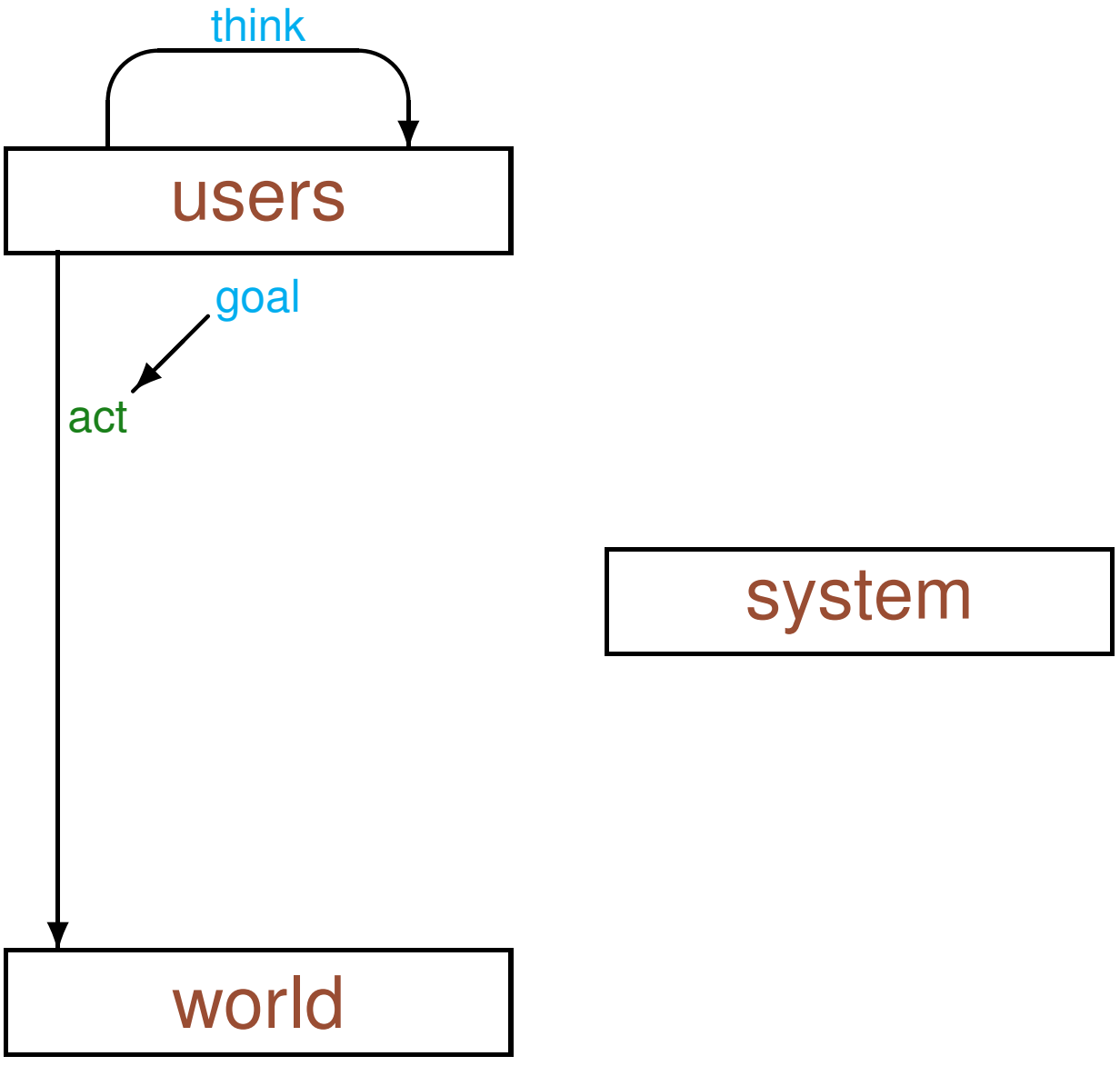
HCI: FM Forward Perspective



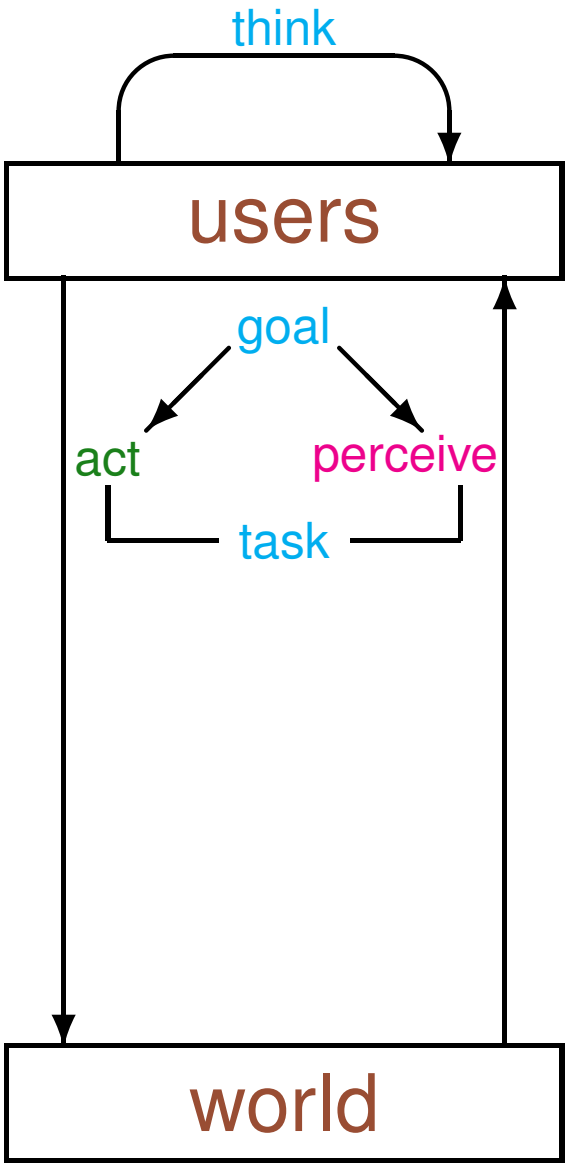
system

world

HCI: FM Forward Perspective

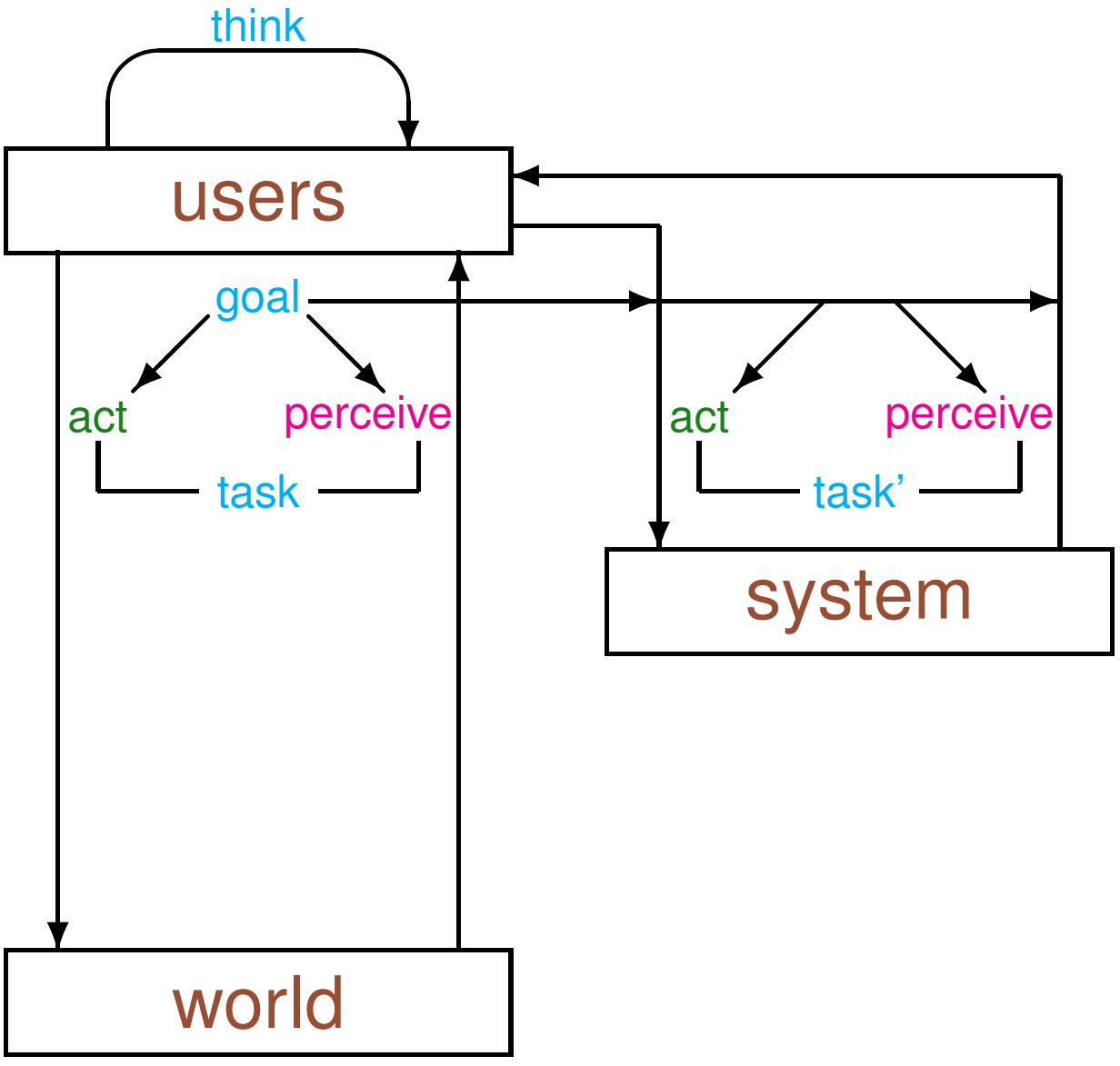


HCI: FM Forward Perspective

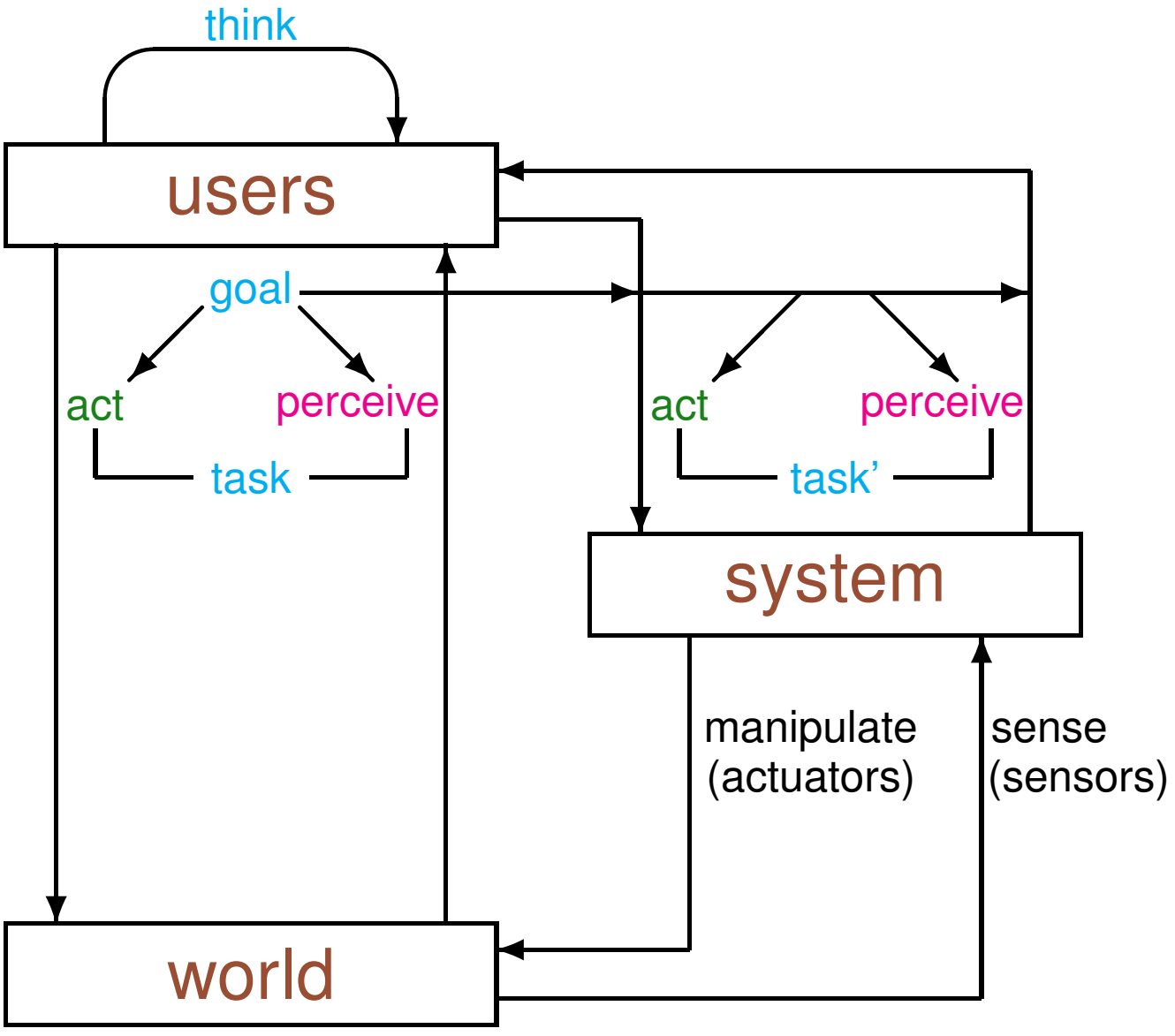


system

HCI: FM Forward Perspective



HCI: FM Forward Perspective



Back to 'Backward Perspective'

There are two problems with the forward perspective

Back to 'Backward Perspective'

There are two problems with the forward perspective

1. a behaviour observed in real-life
   is not predicted by the model

Back to 'Backward Perspective'

There are two problems with the forward perspective

1. a behaviour observed in real-life
   is not predicted by the model
   backward: data from real-life to expand the original behaviour (elaborative mining)
   $\Longrightarrow$ may increase non-determinism

Back to 'Backward Perspective'

There are two problems with the forward perspective

1. a behaviour observed in real-life
   is not predicted by the model
   backward: data from real-life to expand the original behaviour (elaborative mining)
   $\Longrightarrow$ may increase non-determinism
2. a behaviour predicted by the model
   has never been observed

Back to 'Backward Perspective'

There are two problems with the forward perspective

1. a behaviour observed in real-life
   is not predicted by the model
   backward: data from real-life to expand the original behaviour (elaborative mining)
   $\Longrightarrow$ may increase non-determinism
2. a behaviour predicted by the model
   has never been observed
   backward: data from real-life is used to remove part of the behaviour (refinement mining)
   $\Longrightarrow$ reduce non-determinism

The OSS Bazar

E. S. Raymond, *The Cathedral and the Bazar*, O'Really and Associates, 1999

The cathedral-like rigorous and well-documented proprietary software development is contrasted with the 'bazar-like', apparently chaotic OSS development.

The OSS Bazar

E. S. Raymond, *The Cathedral and the Bazar*, O'Really and Associates, 1999

The cathedral-like rigorous and well-documented proprietary software development is contrasted with the 'bazar-like', apparently chaotic OSS development.

L. S. Barbosa, A. Cerone, A. K. Petrenko, S. A. Shaikh, *Certification of OSS Software: A role for formal methods?* Comp. Syst. Sci. Eng. 25(4), 2010

Answer: Backward Perspective

- forward perspective very challenging in the OSS context due to the absence of rigorous verification process and systematic testing and to the heterogenous community involved in the development process

Answer: Backward Perspective

- forward perspective very challenging in the OSS context due to the absence of rigorous verification process and systematic testing and to the heterogenous community involved in the development process
- backward perspective may provide an effective way to use formal methods to assist the re-engineering process of running code: apply principles and calculi in the reverse direction, from concrete to abstract models, for understanding and documenting OSS implementations

Backward Techniques in OSS

- architectural reconstruction exploits open access to source code to extract relevant coordination information (e.g. program analysis adapted to recover info & patterns $\Longrightarrow$ automatically $\Longrightarrow$ orchestration language)

Backward Techniques in OSS

- architectural reconstruction exploits open access to source code to extract relevant coordination information (e.g. program analysis adapted to recover info & patterns $==$ automatically $\Longrightarrow$ orchestration language)
- type reconstruction to understand programs written in untyped or weakly typed languages

Backward Techniques in OSS

- architectural reconstruction exploits open access to source code to extract relevant coordination information (e.g. program analysis adapted to recover info & patterns $\Longrightarrow$ automatically $\Longrightarrow$ orchestration language)
- type reconstruction to understand programs written in untyped or weakly typed languages
- logic mining is a semi-automatic extraction of domain-specific rules from code to improve the results of data reverse engineering

Backward Techniques in OSS

- architectural reconstruction exploits open access to source code to extract relevant coordination information (e.g. program analysis adapted to recover info & patterns $==$ automatically $\Longrightarrow$ orchestration language)
- type reconstruction to understand programs written in untyped or weakly typed languages
- logic mining is a semi-automatic extraction of domain-specific rules from code to improve the results of data reverse engineering
- documentation analysis use a broad range of mining technique to extract data (e.g. requirements) from OSS repositories

Impact Beyond Certification

and Beyond Formal Methods

Documentation Analysis

- extract requirements

Impact Beyond Certification

and Beyond Formal Methods

Documentation Analysis

- extract requirements
- analysis of contributors and community activities (text mining)

Impact Beyond Certification

and Beyond Formal Methods

Documentation Analysis

- extract requirements
- analysis of contributors and community activities (text mining)
- analysis of learning process (process mining)

Impact Beyond Certification

and Beyond Formal Methods

Documentation Analysis

- extract requirements
- analysis of contributors and community activities (text mining)
- analysis of learning process (process mining)
- probabilistic learning of big code (machine/deep learning) to produce statistically likely solution to problems hard to solve with FM: program synthesis, code property prediction, code deobfuscation

Thank you Luís!

It has always been a pleasure to work with you.

Thank you Luís!

It has always been a pleasure to work with you.

And also thank you for the inspiration I got from you and for your great friendship.