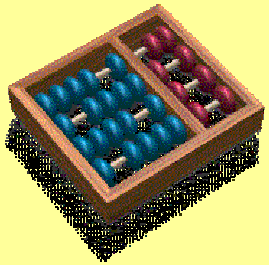


INSTITUTO DE
COMPUTAÇÃO

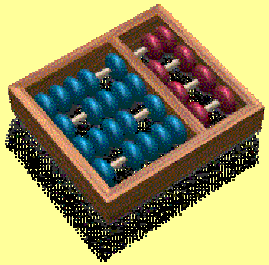
Ceset

Centro Superior de Educação Tecnológica

UNICAMP

DSN 2004
INTERNATIONAL CONFERENCE ON DEPENDABLE SYSTEMS AND NETWORK

JUNE/2004



INSTITUTO DE COMPUTAÇÃO

Ceset

Centro Superior de Educação Tecnológica

Architecture-based Strategy for Interface Fault Injection

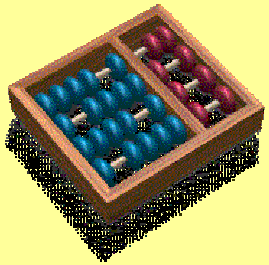
Authors: Eliane Martins

Regina Lúcia de Oliveira Moraes

State University of Campinas - UNICAMP

Brazil

UNICAMP



INSTITUTO DE
COMPUTAÇÃO

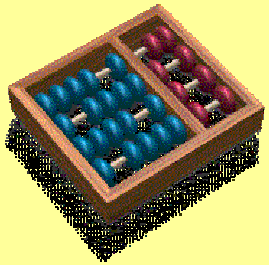
Ceset

Centro Superior de Educação Tecnológica

UNICAMP

Contents

- Motivation
- Fault Injection
- Jaca
- Architectural View
- The Strategy
- Future Works



INSTITUTO DE COMPUTAÇÃO

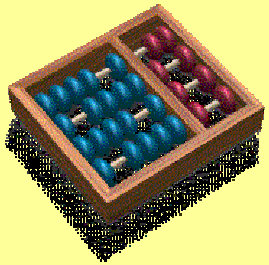
Ceset

Centro Superior de Educação Tecnológica

UNICAMP

Motivation

- Systems as a combination of several components
- Helps to attend the increasing pressures to reduce time and money
- A good solution for system's architecture increases the system's quality



INSTITUTO DE COMPUTAÇÃO

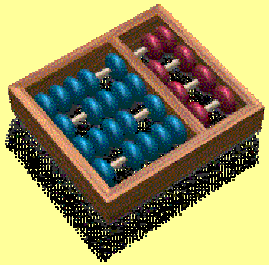
Ceset

Centro Superior de Educação Tecnológica

UNICAMP

Motivation

- Each component's implementation needs to behave in accordance with its specification
- Malfunctioning in the interaction among components can compromise the overall system's quality
- Fault Injection can be a valuable approach to component-based system's testing



INSTITUTO DE COMPUTAÇÃO

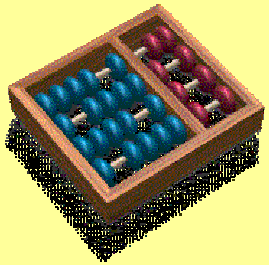
Ceset

Centro Superior de Educação Tecnológica

UNICAMP

Fault Injection

- Fault Injection
  - faults (or errors) are deliberately introduced into a system
  - useful to validate error-handling mechanisms
  - useful to assess system behavior when its components fails
- We use sw-implemented fault injection
  - faults are introduced during runtime
  - faults represent failure modes of components (internal or external to the system)



INSTITUTO DE COMPUTAÇÃO

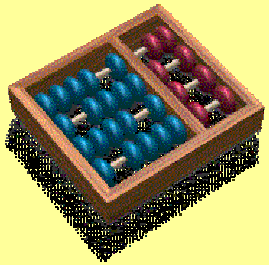
Ceset

Centro Superior de Educação Tecnológica

UNICAMP

Jaca

- Software Fault Injection Tool
- Written in Java language
- Uses computational reflection implemented by Javassist
- Does not need the source code of the system under test
- Can inject high level faults in an object-oriented system written in Java language



INSTITUTO DE COMPUTAÇÃO

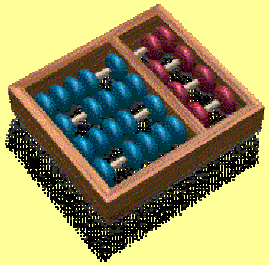
Ceset

Centro Superior de Educação Tecnológica

UNICAMP

Problems to Solve

- Which components to inject?
- Where to inject the faults?
- Which error models to use?
- When to inject the faults?
- How to inject them?

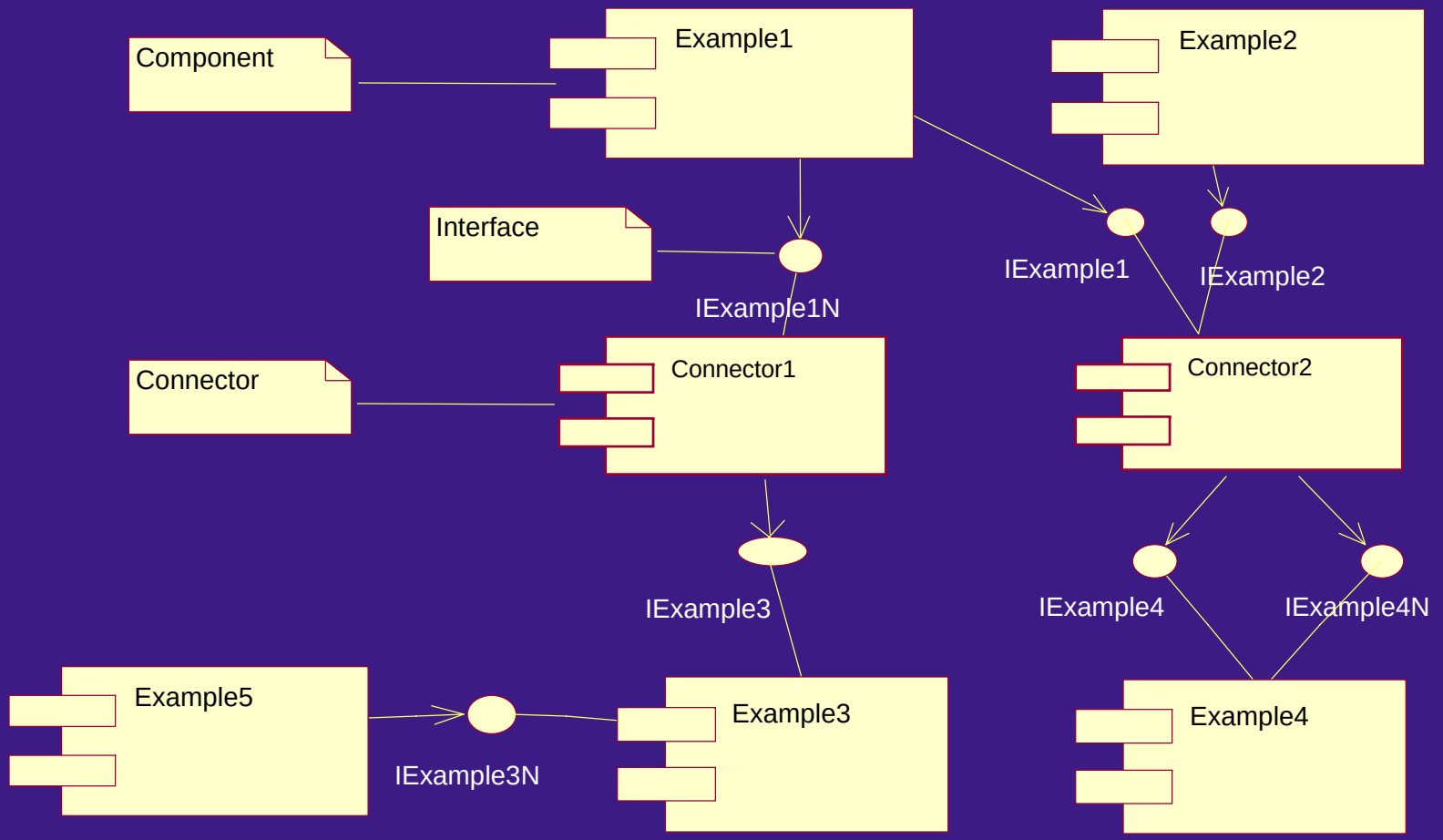


INSTITUTO DE COMPUTAÇÃO

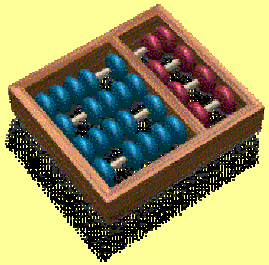
Ceset

Centro Superior de Educação Tecnológica

Architectural View



UNICAMP



INSTITUTO DE COMPUTAÇÃO

Ceset

Centro Superior de Educação Tecnológica

The Strategy

A Strategy based on Risk

- Aimed to answer the following questions:
  - Which components to injec
  - Where to inject

New Component

Changed Component

Upstream Dependency

Downstream Dependency

Critical

Popular

Strategic

Third Party

Distributed

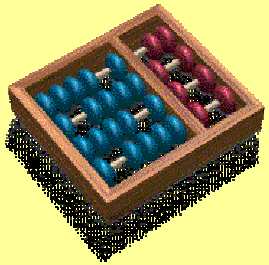
Not Understandable

3. Select Op

4. Generate

nts

UNICAMP



INSTITUTO DE COMPUTAÇÃO

Ceset

Centro Superior de Educação Tecnológica

The Strategy

➢Other questions:

- Which error models to use

5. Define Error Model

- **Ballista's Robustness Testing**
- **Integration Faults**

- When to inject the faults

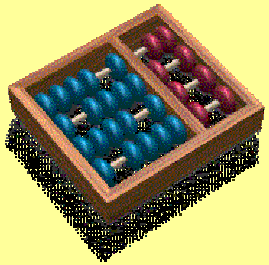
6. Decide Temporal Characterization

- **Randomly selected among permanent, transient and intermittent faults**

- How to inject

Jaca Tool

UNICAMP



INSTITUTO DE COMPUTAÇÃO

Ceset

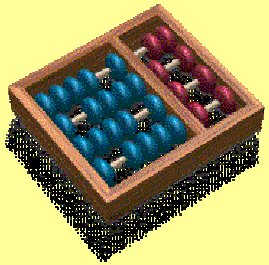
Centro Superior de Educação Tecnológica

UNICAMP

Future Works

Open Problems

- How many criteria must be satisfied?
- How should these criteria be weighted?
- How should each factor be quantified?
- How should successors and predecessors be determined?
- What error model should be selected?
- How can good controlability and observability be achieved?



INSTITUTO DE
COMPUTAÇÃO

Ceset

Centro Superior de Educação Tecnológica

UNICAMP

Thank You for Coming!

Questions and Suggestions

eliane@ic.unicamp.br

regina@ceset.unicamp.br