Robustness and Exception Handling

“The degree to which a system or component can function correctly in the presence of invalid inputs or stressful environmental conditions.”

(ISO/IEC/IEEE 24765:2017 Systems and software engineering — Vocabulary)

One mechanism that supports robustness is exception handling → Equipping the system with the capabilities to tackle classes of abnormal situations

The need of exceptions emerges from the desire of modularizing software, separating concerns into components that interact

Current MAS architectures and methodologies fall short in addressing robustness in a systematic way

No mechanisms for exception handling, as is for programming languages (e.g. Java), or in the actor model (e.g. Akka)

Responsibility in Exception Handling

Two important aspects of exception handling:

- Two parties: the former is responsible for raising an exception, the latter responsible for handling it
- It captures the need for some information from the former to the latter that allows coping with the exception

Since MAS organizations are built upon responsibilities, they are suited to encompass an exception handling mechanism

Extending JaCaMo

Recovery Strategy encodes when and how a given exception is to be raised and handled within the organization

Notification Policy specifies when the exception must be raised

Throwing Goal denotes the organizational goal of raising the exception

Exception Spec encodes the kind of information to be produced by the agent raising the exception

Key features of many organizational models:

- Functional decomposition of the organizational goal
- Normative system

Norms shape the scope of the responsibilities that agents take when joining the organization → What agents should do to contribute to the achievement of the organizational goal

Handling Policy specifies a way in which the exception must be handled

Catching Goal captures the course of action to follow for handling the exception and possibly remediate

Agents are held to explicitly take responsibility for throwing and catching goals

Example: House Building

Handling Policy:

- errorCode (flooding)
- notificationPolicy (must_notify_when)
- handlingPolicy (inspection_companions)

Catching Goal:

- site prep contractor's subcontractor
- site prep engineer
- house owner

Handling Policy:

- errorCode (flooding)
- notificationPolicy (must_notify_when)
- handlingPolicy (inspection_companions)

Catching Goal:

- site prep contractor
- site prep engineer
- house owner

Code of the site prep contractor agent:

1 = obligation(Ag, _ , done(site_prepared, Ag))
2 my_name(Ag)
3 + obligation(Ag, _ , done(site_prepared, Ag))
4 goalAchieved(site_prepared)
5 - site_prepared
6 - notify_problem
7 fail
8 = obligation(Ag, _ , done(notify_problem, Ag))
9 my_name(Ag) & // the site is flooded
10 - throw_exception(site_preparation_exception, errorCode(flooding))
11 goalAchieved(notify_problem)
12 goalAchieved(notify_site_problem)
13 obligation(Ag, _ , done(notify_site_problem, Ag))
14 my_name(Ag) & // the site is flooded
15 - throw_exception(site_preparation_exception, errorCode(archaeologicalRemains))
16 = obligation(Ag, _ , done(notify_site_problem, Ag))
17 goalAchieved(notify_site_problem)
18 goalAchieved(notify_site_problem)

Code of the engineer agent:

1 = obligation(Ag, _ , done(site_prepared, Ag))
2 my_name(Ag)
3 + obligation(Ag, _ , done(site_prepared, Ag))
4 goalAchieved(site_prepared)
5 - site_prepared
6 - inspectSite
7 fail
8 = obligation(Ag, _ , done(inspectSite, Ag))
9 my_name(Ag) & // the site is flooded
10 - throw_exception(site_preparation_exception, errorCode(archaeologicalRemains))
11 obligation(Ag, _ , done(notify_site_problem, Ag))
12 goalAchieved(notify_site_problem)
13 goalAchieved(notify_site_problem)
14 obligation(Ag, _ , done(notify_site_problem, Ag))
15 my_name(Ag) & // the site is flooded
16 - throw_exception(site_preparation_exception, errorCode(archaeologicalRemains))
17 = obligation(Ag, _ , done(notify_site_problem, Ag))
18 goalAchieved(notify_site_problem)
19 goalAchieved(notify_site_problem)

The source code of the extension together with some examples is available at: http://di.unito.it/noiseexceptions
A video presentation of the demonstration is available at: http://di.unito.it/aamas2021demo