symudoc

Release v1.0.0

Dec 03, 2020

Contents

CHAPTER 1

Welcome to Symu's documentation

Symu is a C# opensource framework for **organization simulation**. You can develop your own application to run virtual experiments with organizations and capture their dynamic behaviors, their evolutions.

Some useful links

- Website : symu.org
- Code : github.symu.org
- Issues : github.symu.org/issues
- Twitter : symuorg

1.1 Organization simulation

Symu models groups and organizations as complex systems. It captures the variability in human, technological and organizational factors through heterogeneity in information processing capabilities, but also in knowledge and resources. Also, the non-linearity of the model generates complex temporal behavior due to dynamic relationships among agents.

Symu implements agnostic organizations as social groups to target the most general use cases. Unlike other simulators, symu has cognitive models already implemented and completely configurable to adapt it to your scenarios. It also has advanced tasking and messaging management.

1.2 How it works

Symu models groups and organizations as complex systems and captures the variability in human, technological and organizational factors through heterogeneity in information processing capabilities, knowledge and resources.
 The non-linearity of the model generates complex temporal behavior due to dynamic relationships among agents.

1.3 What it is

Symu, is a multi-agent system, time based with discrete events, for the co-evolution of agents and socio-cultural environments. Agents are decision-making units and can represent various levels of analysis such as individuals, groups or organizations.
 Agents are autonomous, rationally bounded and tasks based.
 They interact simultaneously in a shared environment that interacts in turn with the agents, via asynchronous messages.

1.4 Project infos

CHAPTER 2

Models

Simulation is based upon models. This section defines the main models used by Symu.

2.1 Simulation

2.1.1 How it work

The simulation method is known as an "agent-based model" or multiagent system which is a type of computer simulation approach that is used in biological sciences and the social sciences as a way to model systems more holistically. The agents in the model could be cells, teams, individuals, or they could be companies. The agents interact with one another. In this approach, properties can emerge of the system during the simulation. These result from many discrete local interactions between the agents that add up to a new system state.

See also:

Agent

2.1.2 Engine

The engine of this simulator offers the possibility to performe or multiple iterations. Multiple iterations is useful in a Monte Carlo approach. This method is aiming to calculate an approximate numerical value using random methods.

2.1.3 Scenarios

To run a simulation, you must at least define one scenario. it will basically allow to define which parameter allows to stop the simulation and for which value. The pre-defined scenarios are: * Time based * Tasks based * Messages based You can overload your own scenario. A simulation may be define with multiple scenarios. In that case, the simulation will stop when all the scenarios will be stopped.

2.1.4 Events

To add variability, you can schedule events during the simulation. You will find by design one shot, cyclical and random events.

Examples

You can add a new agent every 100 steps, or randomly add an event that affect the beliefs of the agents.

You can only use scenarios if you choose. But the scenarios and the events are different by design. The scenario is an agent, has its own life cycle, interacts with other agents via asynchronous messages; where events is a simple class that can be used directly by the agent via an EventHandler.

At least one scenario is required to run a simulation, then you choose what best suits your needs for variability between scenarios and events.

See also:

SymuScenariosAndEvents example

2.1.5 Models

`Symu` is a time-based. It contains a discrete event schedule on which you can schedule various agents to be called at some specific time. But it is ont space-based, it has no 2D nor 3D features to localize agents in the space. Of course, Simulation is at the heart of this framework. It is based on two asynchronous models:

- Messaging between agents
- Tasks for each agent

Messaging model

Messaging is the only way agents communicate to each other. So messaging model represents the interaction mechanism of agents. This is one of the means of ensuring the autonomy of agents. A message can have different types: it can be a phone call, a meeting, an email, . . .

Each type of message can be configure via templates.

Note: to be done

Message content

Message content is part of the cognitivearchitecture.

Messaging is the only way agents communicate to each other. So messaging model represents the interaction mechanism of agents. This is one of the means of ensuring the autonomy of agents. A message can have different types: it can be a phone call, a meeting, an email, . . . Message content helps you define the content of a message : knowledge, beliefs, referral, . . .

Knowledge content

Each agent may be able to send or not knowledge (or information) and to receive or not knowledge.

Example

You can use a book to learn new knowledge. As an agent, a book is able to send knowledge, but not able to receive knowledge.

Interaction characteristics

Interaction characteristics is part of the cognitivearchitecture.

index:: Messaging is the only way agents communicate to each other. So messaging model represents the interaction mechanism of agents. This is one of the means of ensuring the autonomy of agents. A message can have different types: it can be a phone call, a meeting, an email, ... Interaction characteritics model defines the way an agent will interact with other agents. There are two main aspects of the model that you can configure:

- limit
- cost

Limit messages

You can limit the total number of index:: messages an agent can send or received during a step; you can also specify the number of messages sent or the number of receptions per agent per step. It can be useful when you want to test the limitation of interaction between agents, whatever the interaction.

Cost of the messages

Sending or receiving a message can have a cost that depends of the type of the message. The cost may be a fraction of the capacity of an agent. For example, for an agent with a capacity of 1, going to a meeting of two hours, the cost to send the message may be equal to 0 (sending invitations) and the cost to receive the message may be equal to 0.25 (doing the meeting).

Tasking model

In Symu, agents are task-based. A task is an action with a cost. Some of them don't perform task, such as a book or a static website; others can perform task such as workers. You can specify if the agent can perform task on weekends or not, that can be useful when your organization is an enterprise.

Tasks and performance

Tasks and performance is part of the cognitivearchitecture.

Tasking model

In Symu, agents are task-based. A task is an action with a cost. Some of them don't perform task, such as a book or a static website; others can perform task such as workers. You can specify if the agent can perform task on weekends or not, that can be useful when your organization is an enterprise. There are also two main aspects of the model that you can configure:

- limit
- capacity

Limit tasks

You can limit the total number of tasks an agent can perform during the entire simulation. You can also limit the number of simultaneous tasks an agent is performing. If you want to avoid multi-tasking, you can set the limit to one. It is also an easy way to create a pull-system. When you allow multi-tasking, you can define the impact of context switching between tasks on capacity. In that case, multiple tasks may be in progress at the same time.

Cost of task and capacity of the agent

By default, agent's capacity is initialized every step to 1. This method can be overload to add more sophisticated models. This capacity can be less than 1. It depends on the murphies' setups. This capacity is used to perform tasks. So, a task has a cost. Each time an agent has done a task, the capacity of the agent is decreased of the cost of the task.

Learning model

Agents have the capacity to learn new knowledge or information during the simulation. There are different means to learn new knowledge:

- learning from a source of information
- by interacting with another agent
- by doing by itself.

This model is mainly defined by a rate of learning (learn rate and learn by doing rate). It defines how quickly an agent will learn new knowledge when interacting with other agents. With a rate of 0.01, if an agent has initially a knowledge of 0.5, after a learning, its knowledge will be 0.5 +0.01 = 0.51.

Learning by doing

A special case is when an agent is doing by itself and gain knowledge. For that, agent must have a minimum of initial knowledge to do it by itself (knowledge threshold for doing) and it has a cost (cost factor), the associated task will take longer than if he already knew it, that is the cost of learning.

2.2 Agent

The agents in the model could be cells, teams, individuals, or they could be companies. The agents interact with one another. In an agent-based simulation, you define agents, their characteristics like their behaviours, what information they can know and what state of the environment they inherit before they can act.

IN that way, each agent as a cognitivearchitecture/cognitivearchitecture which allow to set up agent's models. This model is inspired by Construct specifications[1].

Each model can be turn off or on. When the model is on, you can choose the percentage of agents that are impacted by the model. If 0 is chosen, any agent will be impacted, if 1 is chosen every agent will be impacted.

2.2.1 Agent's models

- Learning model
- Forgetting model
- Influence model
- Knowledge model
- Belief model

Learning Model

Agents have the capacity to learn new knowledge or information during the simulation. There are different means to learn new knowledge:

- learning from a source of information
- by interacting with another agent
- by doing by itself

Forgetting model

The counterpart of learning is forgetting. Agents may forget knowledge or information if they are not solicited during the simulation. Using a bit of knowledge during a step is enough to be sure that this bit will not be forget today.

See also:

cognitivearchitecture/internalcharacteristics

Influence model

This model define how an agent will reinforce its belief or change its belief from influencer. Agents can accumulate beliefs during the simulation. INfluence model is associated with the risk aversion that prevent an agent from acting on a particular belief.

See also:

Knowledge and beliefs

Knowledge model

Note: to be done

---

[1] "Specifying agents in Construct", Carley, 2007

Beliefs model

Note: to be done

2.3 Interaction patterns

Interaction patterns is part of the cognitivearchitecture.

Symu is a multi-agent system for the co-evolution of agents and socio-cultural environments with an interaction model between agents. Each agent defines the way the agent interacts with other agents. Certain types of non-relational agents such as a database are not part of the sphere. For those who are part of the interaction sphere, the sphere is computed depending on different parameters.

2.3.1 Interaction sphere

It is based on different parameters:

- Relative knowledge
- Relative activities
- Relative beliefs
- Social demographics

Each parameter has an associated weight to calculate the index:: homophily factor.

To interact with another agent, an agent filter the interaction sphere with an interaction strategy based on:

- Relative knowledge
- Relative activities
- Relative beliefs
- Social demographics
- Homophily

If the interaction value between two agents is below the average of the interaction sphere it is considered as a **new interaction**. In that case, when the two agents try to interact, they go through the **new interaction process**.

2.3.2 Sphere initialization

This model defines the way the interaction sphere is initialized and can change over time. The interaction sphere may be initialized in two ways: similarity matching and random generation.

Similarity matching

In that case, the interaction sphere is generated based on known informations about agents (knowledge, activities, beliefs, socio-demographics).

Random generation

This option can be usefull when you don't want to define nowledge, beliefs, … It is based on parameters such as min/max density of the sphere. This specifies the proportion of agents who can appear in a given agent's interaction sphere. If an agent appears in the interaction sphere, its weigth is randomly generated.

2.3.3 Sphere evolution

During the simulation, the interaction sphere is updated with the updated agent's informations. As the action is time consuming and based on the fact that knowledge, beliefs, … don't evolve too frequently, you can adjust the frequency of update every day, every week, every month, enevry year.

The sphere size may change over time if new agents are on boarding or agents are leaving the simulation.

New interaction process

To be able to initiate a new interaction, an agent must have the right to do it (parameter AllowNewInteractions). The number of new interactions per step may be limited (parameters LimitNumberOfNewInteractions and MaxNumberOfNewInteractions). When an agent receive a demand for a new interaction, he answer randomly (depending on the parameter ThresholdForNewInteraction).

2.4 Knowledge and beliefs

Knowledge and beliefs is part of the cognitivearchitecture.

Symu is a multi-agent system for the co-evolution of agents and socio-cultural environments with an interaction model between agents. Each agent defines the way the agent interacts with other agents. certain types of non-relational agents such as a database are not part of the sphere. For those who are part of the interaction sphere, the sphere is computed depending on different parameters

2.4.1 Knowledge model

Agents have the capacity to have initial knowledge, during the simulation (learning model). If an agent has initial knowledge, it is initialized randomly depending on the knowledge level. You can choose the level in a list from no knowledge to full knowledge.

Knowledge is defined by an array of bit of information between 0 and 1. The length of this array is a parameter between 0 to 100.

For example, the knowledge to use Symu can be defined as an array of 20 bits. If you don't know anything about it, the array is filled with 0. If you have a full knowledge of it, it will be filled with 1. Most of the case, the array will be filled randomly with 0/1 if the Binary Knowledge is chosen, or with float between [0; 1] otherwise.

2.4.2 Beliefs model

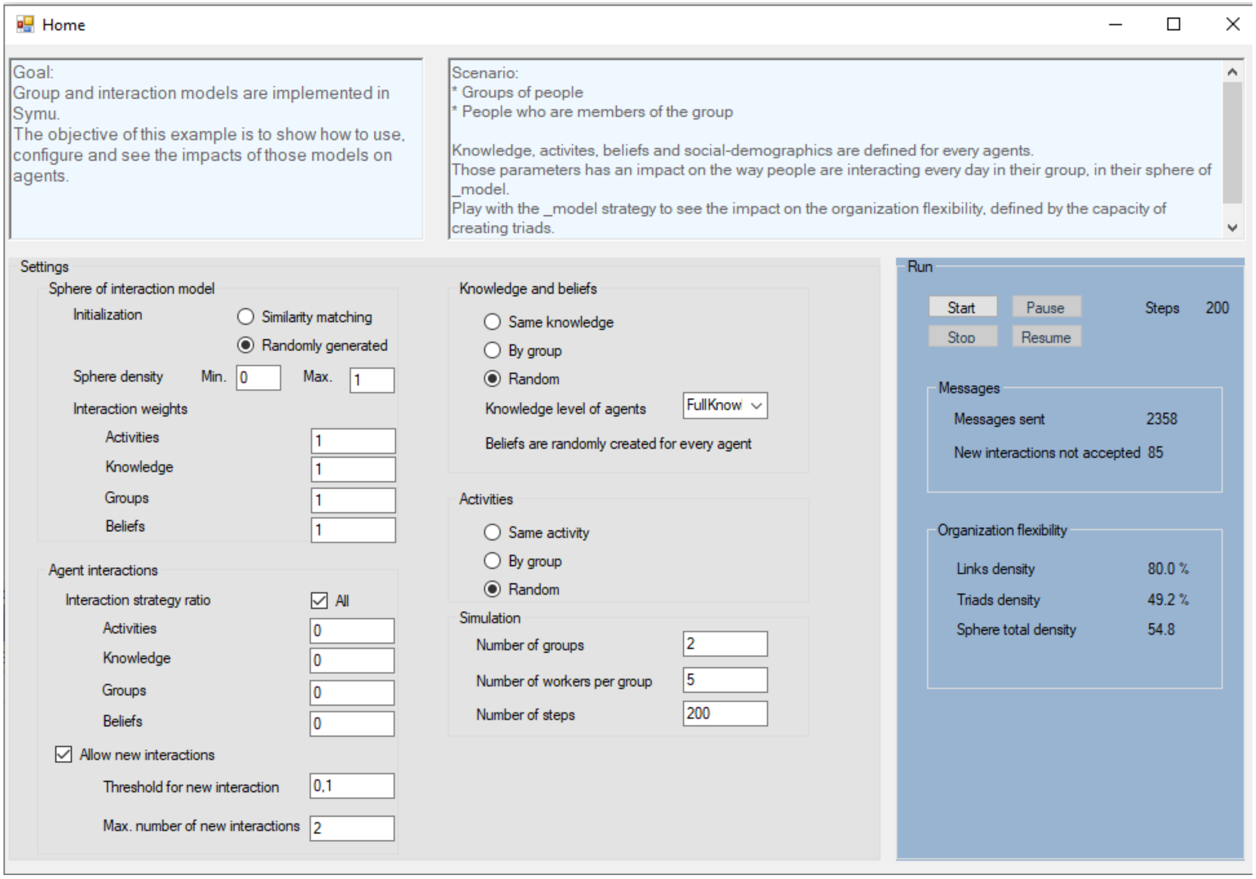
Symu is a multi-agent system for the co-evolution of agents and socio-cultural environments with an interaction model between agents. These interactions are based on the information exchanged but also the beliefs.

CHAPTER 3

Examples Guide

the purpose of the examples is to explore the use of certain models of the framework, understand the impact on performance and how to configure it. So that you will be able to configure models for your own use cases.

3.1 SymuGroupAndInteraction example



Different models are implemented in this simulator. SymuGroupAndInteraction demonstrate the use of one of them: the index:: interaction model.

You can use this example to discover and understand the impact of all the setups on the agents. That will help you setup you own interaction model.

3.1.1 Interaction model

Symu is a multi-agent system for the co-evolution of agents and socio-cultural environments with an interaction model between agents. Find more information about the interaction model.

See also:

Interaction patterns

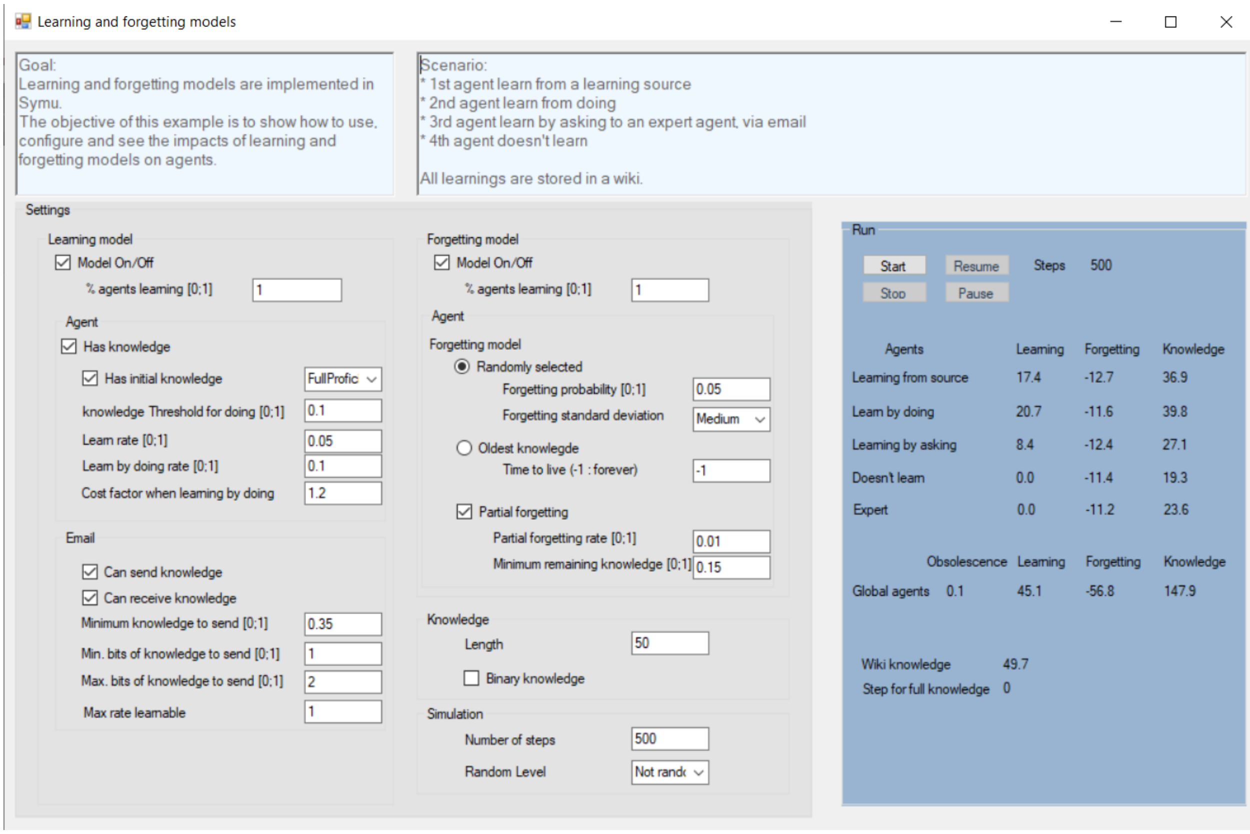
3.1.2 Scenario

- Groups of people
- People who are members of the group

Knowledge, activites, beliefs and social-demographics are defined for every agents. Those parameters has an impact on the way people are interacting every day in their group, in their sphere of interaction. Play with the interaction strategy to see the impact on the organization flexibility, defined by the capacity of creating triads. Try to increase triads with the creation of new interactions.

Find the source code

3.2 SymuLearnAndForget example



Different models are implemented in this simulator. This example demonstrates the use of three of them: index:: knowledge, index:: learning and index:: forgetting.

3.2.1 Knowledge model

Agents can start the simulation with initial knowledge or information.

3.2.2 Learning model

Agents have the capacity to learn new knowledge or information during the simulation. There are different means to learn new knowledge:

- learning from a source of information
- by interacting with another agent
- by doing by itself

3.2.3 Forgetting model

The counterpart of learning is forgetting. Agents may forget knowledge or information if they are not solicited during the simulation. Using a bit of knowledge during a step is enough to be sure that this bit will not be forget today. Forgetting has different modes:

- it could be random
- based on the age of the information

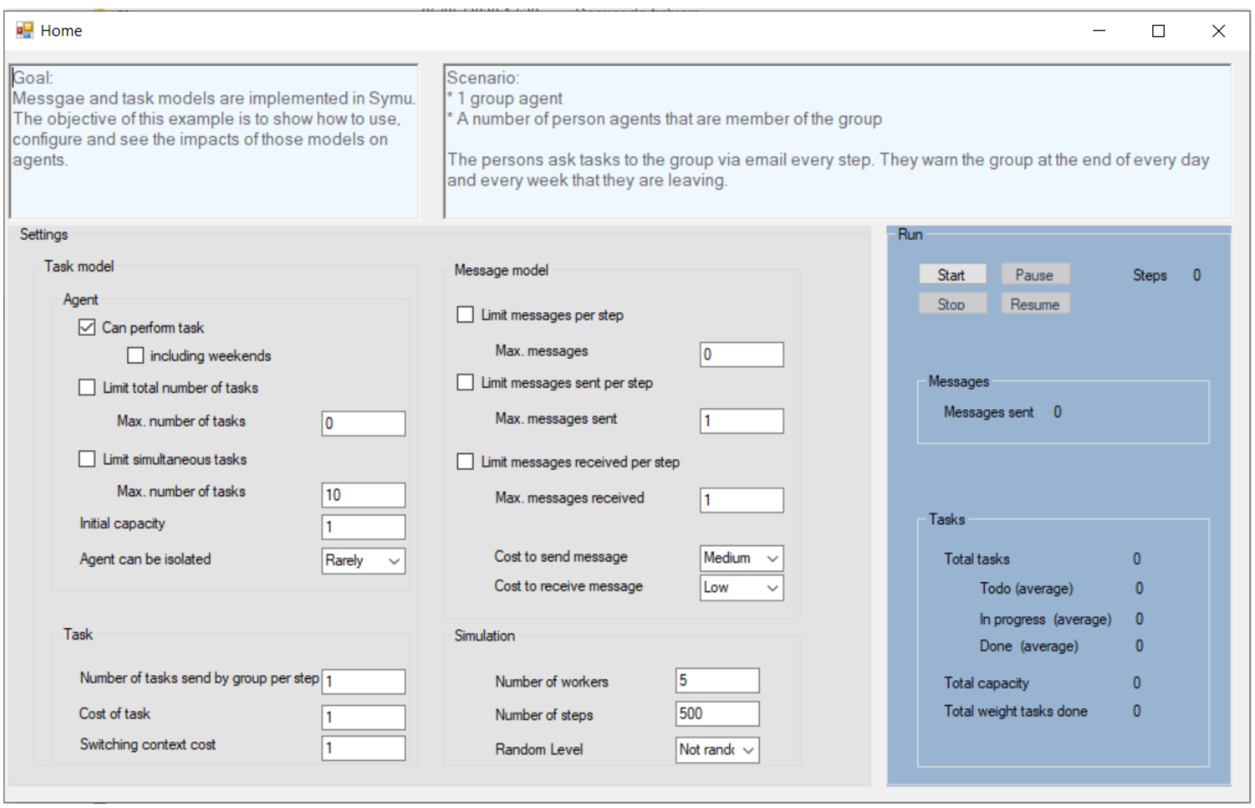
3.2.4 Scenario

You can use this example to discover and understand the impact of all the setups on the agents. That will help you setup you own learning and forgetting models.

1. The first agent learns from a learning source
2. The second agent learns from doing
3. The third one learns by asking to an expert agent
4. The last agent doesn't learn
5. All learnings are stored in a wiki.

Find the source code

3.3 SymuMessageAndTask example



Different models are implemented in this simulator. This example demonstrates the use of two of them: index:: messaging and index:: tasking. We have treated those two models at the same time because both are the engines of the simulation.

3.3.1 Messaging model

Messaging is the only way agents communicate to each other. So messaging model represents the interaction mechanism of agents. This is one of the means of ensuring the autonomy of agents. A message can have different types: it can be a phone call, a meeting, an email, . . .

3.3.2 Tasking model

In Symu, agents are task-based. A task is an action with a cost. Some of them don't perform task, such as a book or a static website; others can perform task such as workers. You can specify if the agent can perform task on weekends or not, that can be useful when your organization is an enterprise.

3.3.3 Scenario

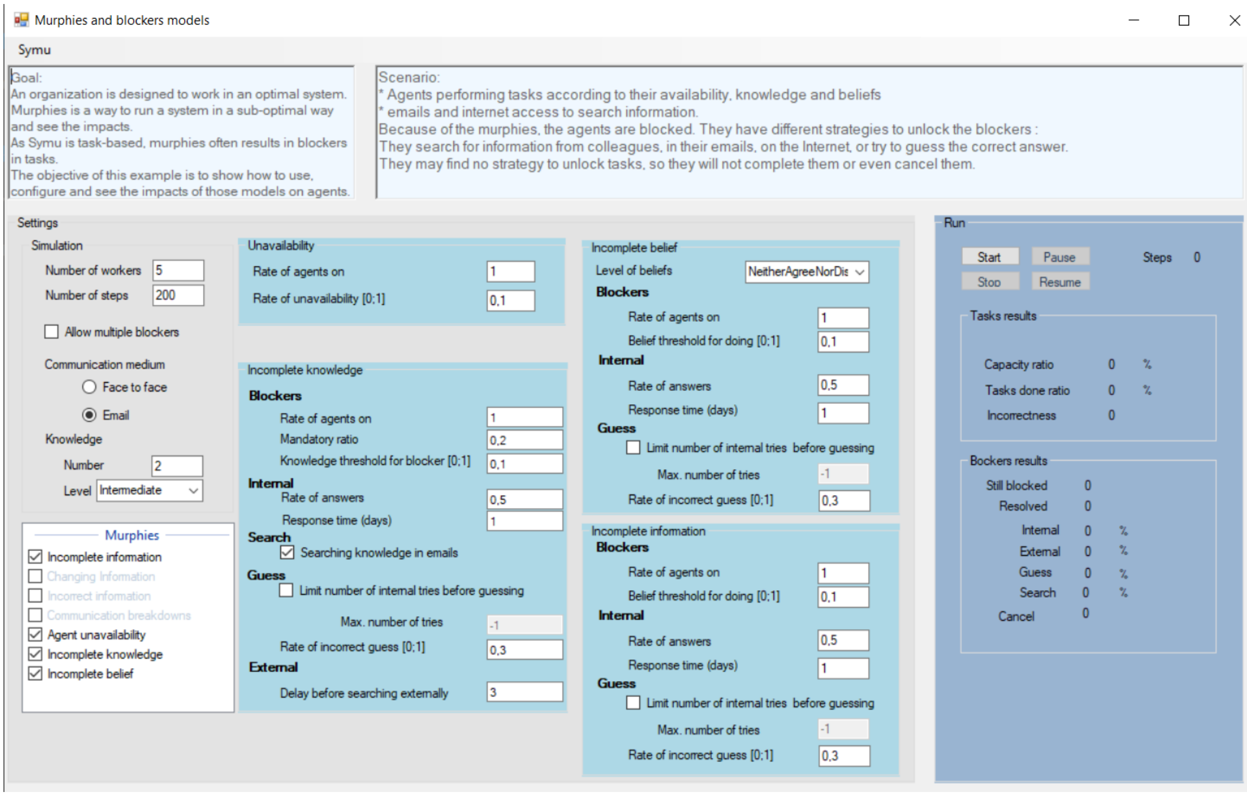
You can use this example to discover and understand the impact of all the setups on the agents. That will help you setup you own messaging and tasking models.

1. 1 group agent
2. A number of person agents who are members of the group

The persons ask tasks to the group via email every step. They warn the group at the end of every day and every week that they are leaving.

Find the source code.

3.4 SymuMurphiesAndBlockers example



An organization is designed to work in an optimal system. Murphy is a way to run a system in a sub-optimal way and see the impacts. As Symu is task-based, murphies often results in blockers in tasks. The objective of this example is to show how to use, configure and see the impacts of those models on agents.

3.4.1 Murphies models

By construction, Symu offers you several possible scenarios for working sub-optimally:

- Unavailability
- Incomplete knowledge
- Incomplete beliefs
- Incomplete information

See also:

../models/murphies

3.4.2 Scenario

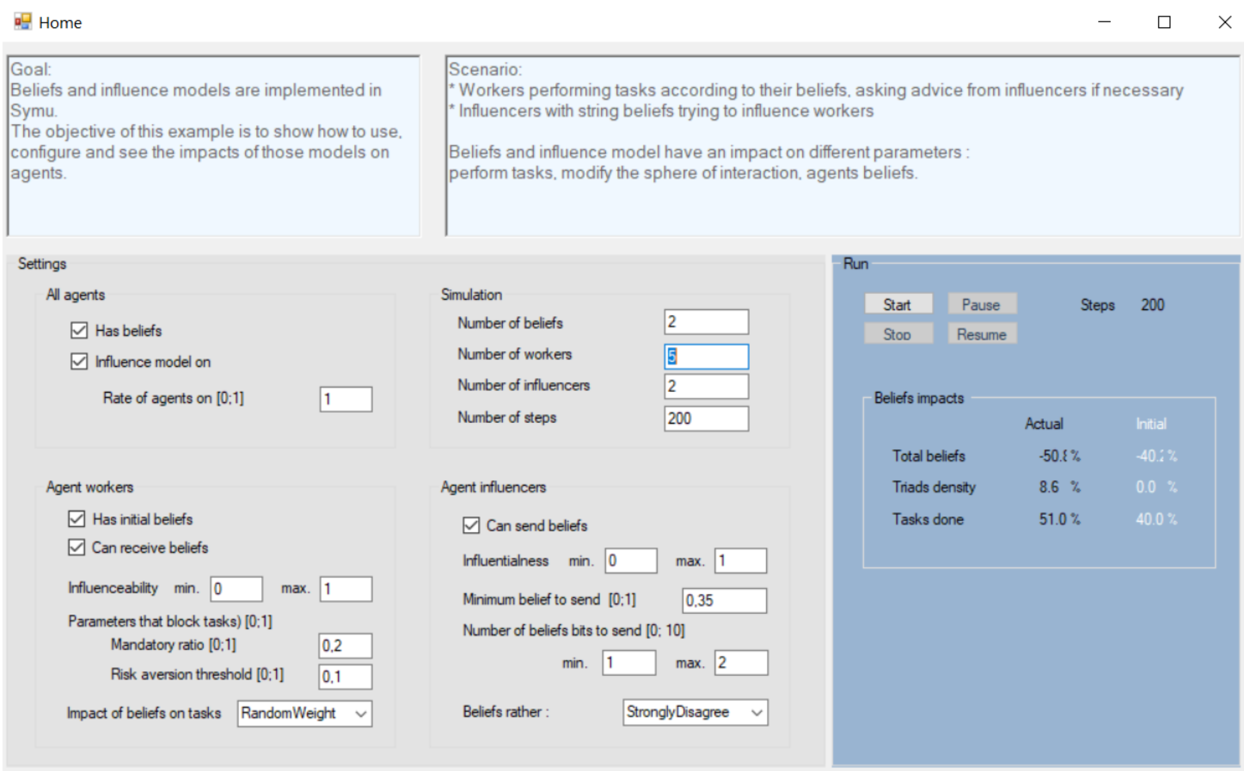
1. Agents performing tasks according to their availability, knowledge and beliefs
2. Emails and internet access to search information.

Because of the murphies, the agents are blocked. They have different strategies to unlock the blockers : They search for information from colleagues, in their emails, on the Internet, or try to guess the correct answer. They may find no strategy to unlock tasks, so they will not complete them or even cancel them.

You can use this example to discover and understand the impact of all the setups on the agents. That will help you setup you own murphies models.

Find the source code.

3.5 SymuBeliefsAndInfluence example



Different models are implemented in this simulator. SymuBeliefsAndInfluence demonstrate the use of two of them: the beliefs and the influence model.

You can use this example to discover and understand the impact of all the setups on the agents. That will help you setup you own interaction model.

3.5.1 Beliefs model

Symu is a multi-agent system for the co-evolution of agents and socio-cultural environments with an interaction model between agents. These interactions are based on the information exchanged but also the beliefs.

See also:

../models/cognitivearchitecture/Knowledgeandbeliefs

3.5.2 Influence model

This model define how an agent will reinforce its belief or change its belief from influencer. .. seealso:: *Agent*

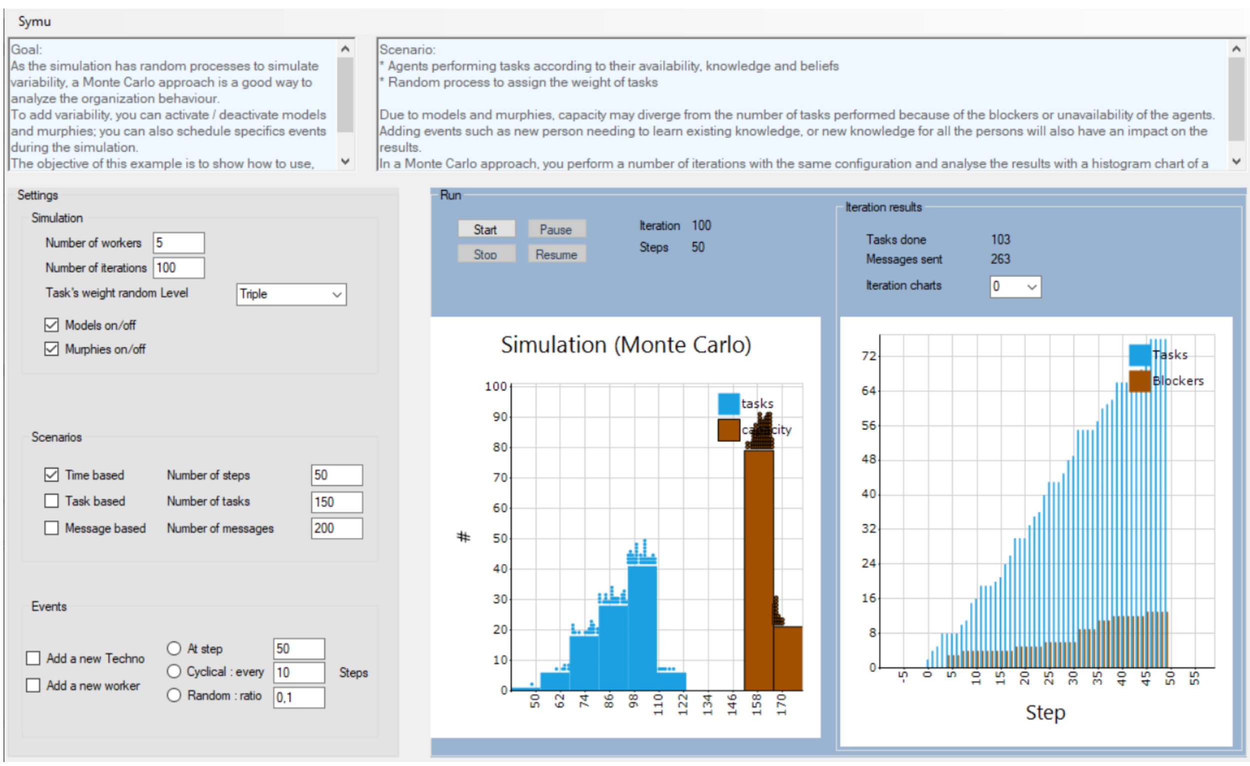
3.5.3 Scenario

- Groups of people
- People who are members of the group

Knowledge, activites, beliefs and social-demographics are defined for every agents. Those parameters has an impact on the way people are interacting every day in their group, in their sphere of interaction. Play with the interaction strategy to see the impact on the organization flexibility, defined by the capacity of creating triads. Try to increase triads with the creation of new interactions.

Find the source code

3.6 SymuScenariosAndEvents example



You can use this example to discover and understand how to use scenarios and events models to run a simulation. As the simulation has random processes to simulate variability, a Monte Carlo approach is a good way to analyze the organization behaviour. To add variability, you can activate / deactivate models and murphies; you can also schedule events during the simulation. The objective of this example is to show how to use, configure and see the impacts of this approach on results.

3.6.1 Scenarios

To run a simulation, you must at least define one scenario. it will basically allow to define which parameter allows to stop the simulation and for which value. There are thress pre-defined scenarios : * Time based * Tasks based * Messages based You can define your own scenario. A simulation may be define with multiple scenarios.

3.6.2 Events

To add variability, you can schedule events during the simulation. You will find by design one shot, cyclical and random events. You can only use scenarios if you choose. But the scenarios and the events are different by design. The scenario is an agent, has its own life cycle, interacts with other agents via asynchronous messages; where events is a simple class that can be used directly by the agent via an EventHandler.

At least one scenario is required to run a simulation, then you choose what best suits your needs for variability between scenarios and events.

See also:

Simulation

3.6.3 Scenario

- Agents performing tasks according to their availability, knowledge and beliefs
- Random process to assign the weight of tasks

Due to models and murphies, capacity may diverge from the number of tasks performed because of the blockers or unavailability of the agents. Adding events such as new person needing to learn existing knowledge, or new knowledge for all the persons will also have an impact on the results. In a Monte Carlo approach, you perform a number of iterations with the same configuration and analyse the results with a histogram chart of a frequency distribution in which the height of the bars are proportional to the result frequencies.

Find the source code

CHAPTER 4

Welcome to the technical guide for developpers

Symu is a core of discrete-event multiagent simulation library in C#. It is designed to be the foundation for custom-purpose organization simulations. It also to provides enough functionality for many simulation needs. Symu implements agnostic organizations as social groups to target the most general use cases. There are three main classes to discover this framework:

- SymuEngine
- SymuEnvironment
- SymuAgent

4.1 SymuEngine

4.2 SymuEnvironment

SymuEnvironment forms the basis for the representation of all environmental dynamics and structure that enables agents inetractions. Symu is a time-based simulator. At each step, SymuEnvironment randomly send messages about the time to the agents. It contains a scheduler. The Schedule is your simulation's representation of time.

4.3 Agent

4.3.1 Agent life cycle

Agent is created, its state is NotStarted. You can setup its cognitivearchitecture at this stage via SetCognitive() method, or by code when unit testing. When launching the simulation, Agent.Start() is called. During the starting phase, a BeforeStart() method is called which is more dedicated to Models via SetModels() method. Models often are based on the CognitiveArchitecture, that why they are define at the latest moment. When the agent's state is started, you can manipulate the default and specific models of the agent.

See also:

for a more functional definition of agent, you can read *Agent*

CHAPTER 5

Indices

- genindex
- search

Index