Mid Sweden University
Department of Information
Technology and Media
Palmius J.

2002-09-26

CASID/AMSIDO-1 Models

1. Background

Most of the following section was previously compiled into a work document called “AMSIDO-1 Models”. That document was mostly written as a reference for the then coming evaluation of the CASID models (see AMSIDO-2, Validation and data collection).

For the update into this document, the contents of the older document was revised and complemented with some input from discussions following the production of the literature overview (see AMSIDO-2, Social Simulations).

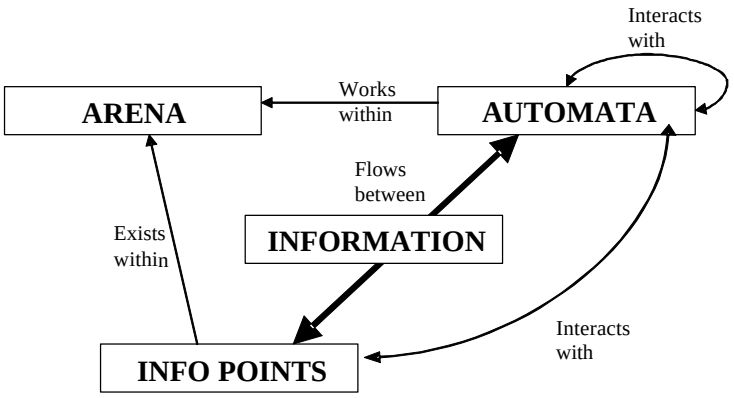
During the fall -01, the AMSIDO-1 project produced a reference implementation of a set of models for simulating information distribution. The results of this implementation was presented in a report in January -02. In that report, a number of models were described textually. The models were also (more or less) implicitly available in the configuration files of the reference implementation. However, the models themselves were never described explicitly in any generally available document.

A few months after the conclusion of AMSIDO-1, the AMSIDO-2 project group found that a structured and explicit description of the available models would be useful. At the time an evaluation and validation of the efforts so far was being initiated and thus a reference for that work was necessary.

The following subsections summarize the *models underlying* the AMSIDO-1 simulation. Note that while the models underlie the simulation, far from all models and parts of the models are implemented. The section "Implementation" contains a discussion about this.

The work in AMSIDO-2 has produced more extensive models than the ones produced in AMSIDO-1. These models are not described in detail here (as they are described in (see AMSIDO-2, Social Simulations), but the last section of this document described the deficiencies found so far in the AMSIDO-1. That section also proposes some extensions that may be parts of what comes out of AMSIDO-2.

2 Overall Model Hierarchy

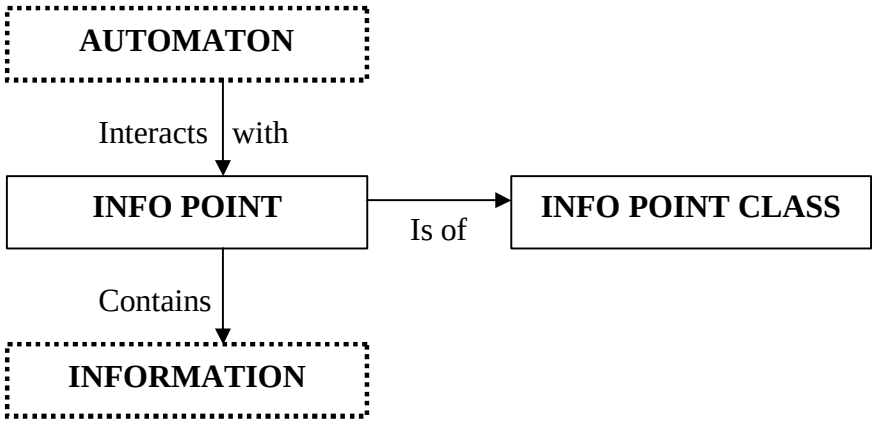


The basics for the AMSIDO-1 model are that it is separated into four distinct "entities" which in their turn are separated into component parts. The entities are:

- **Arena**: The physical structure of the organization modelled in the simulation. This entity contains walls, rooms, doors and similar. The arena provides "game board" in which the other entities are placed and moves.
- **Automata**: The "automata" are representing the persons working within the organization. Each "automaton" is an individual with an organizational belonging, a geographical position within the arena and a number of properties. The automata interact with other automata and with "info points". The automata look for and distribute information.
- **Info point**: The "info points" are the technological artefacts with the purpose to store and distribute information. An info point can as an example be a phone or a bulletin board. The points are stationary on a geographical position within the arena.
- **Information**: The "information" is a representation of the messages flowing within the organization. The information do never exist separately, but is contained (in copies) within the automata and the info points.

Below, the entities are described in more detail.

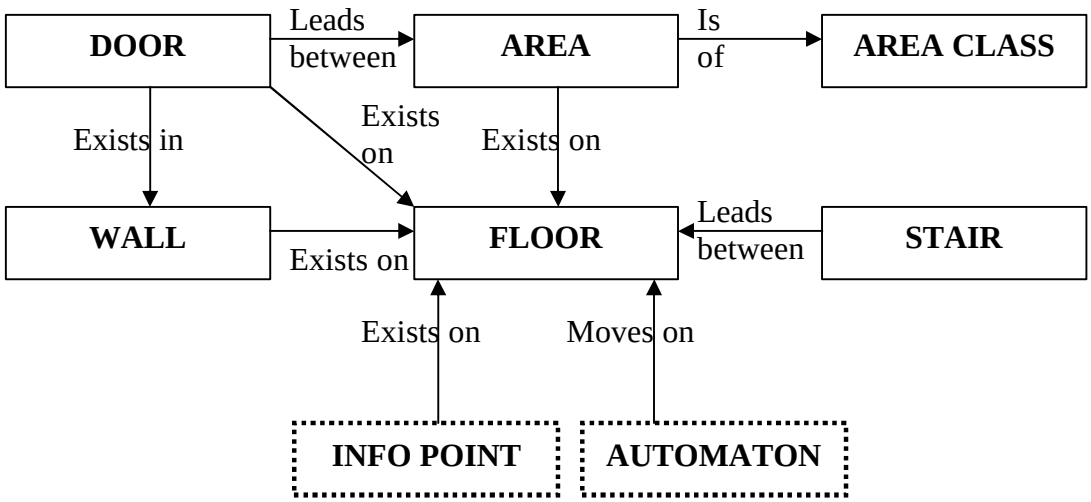
3 Info Point Entity



As mentioned, the "info point" is a technological artefact with the purpose to store and distribute information. Modelwise, the "info point" in itself is not much more than a serial number, a geographical position, a list of contained information, and a reference to an info point class. In other words, the info point is the *specific* item placed in the arena. The info point class describes the *general* properties for a class of info points. The "info point class" can as an example be "phone", while "info point" is "item of class phone, placed on...". The info point class is a set of variables describing the properties relevant for information distribution:

PROPERTY	**DESCRIPTION**
Name	A label describing the class, as an example "phone"
Max receivers	The number of automata that can *get* information simultaneously
Max senders	The number of automata that can *put* information simultaneously
Rec. distance	How far away an automaton can be when getting information
Send distance	How far away an automaton can be when putting information
Rec. speed	How long time it takes (for an automaton) to get an information
Rec. deviation	The variation in the time it takes to get an information
Send speed	How long time it takes (for an automaton) to put an information
Send deviation	The variation in the time it takes to put an information
Index time	How long time it always takes *before* information retrieval can be initiated
Index deviation	The variation in the index time
Index multiple	How much to add to the index time per contained information
Storage capa.	How much information the point can store
Storage time	How long an information is stored in the point
Medium	Which media the point accepts (Audio, video, text...)
Cannot send	Whether the point can send information at all
Cannot rec.	Whether the point can receive information at all
Sent disappear	Whether information is removed when sent

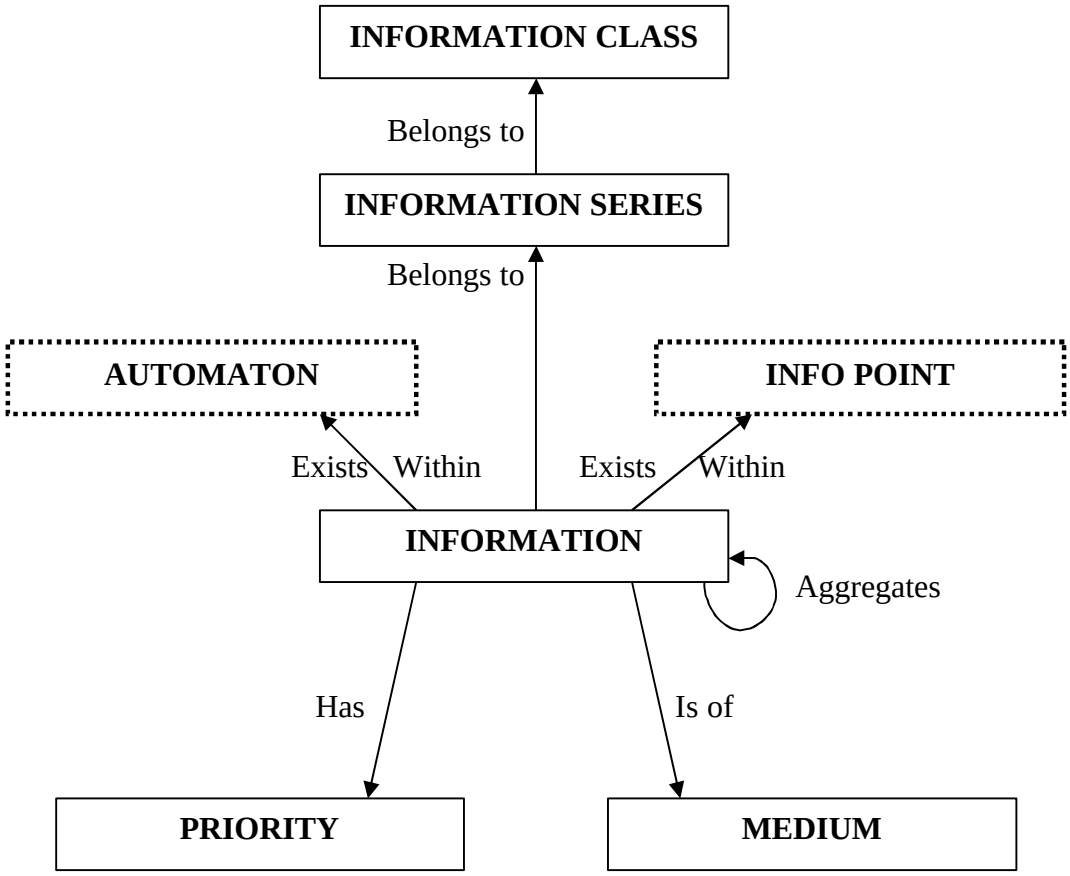
4 Arena Entity



The arena entity describes the physical container for the organization. This is intended to be a one-to-one mapping with a floor plan. The arena consists of five parts which common properties a geographical coordinates:

- **Floor**: The floor is mainly a way to group the other arena parts on "floors" (in the sense of a building having several floors). The floor does not have any properties in itself.
- **Wall**: The wall is a physical delimiter placed upon a floor. It stretches from one 2D coordinate to another 2D coordinate on that floor.
- **Door**: The doors are two-dimensional areas placed on walls. The doors denote points where automata can pass through a wall. Normally the door leads between two "areas".
- **Stair**: The stair leads between two floors. In practise one "real" stair is two "virtual" stairs (one leading from floor A to floor B, and one from B to A).
- **Area**: The area is a room, or part of a room, with an intended purpose. The area is of a specified **area class**. The area class can be "personal office", while the area can be "a personal office placed at coordinates...".

5 Information Entity



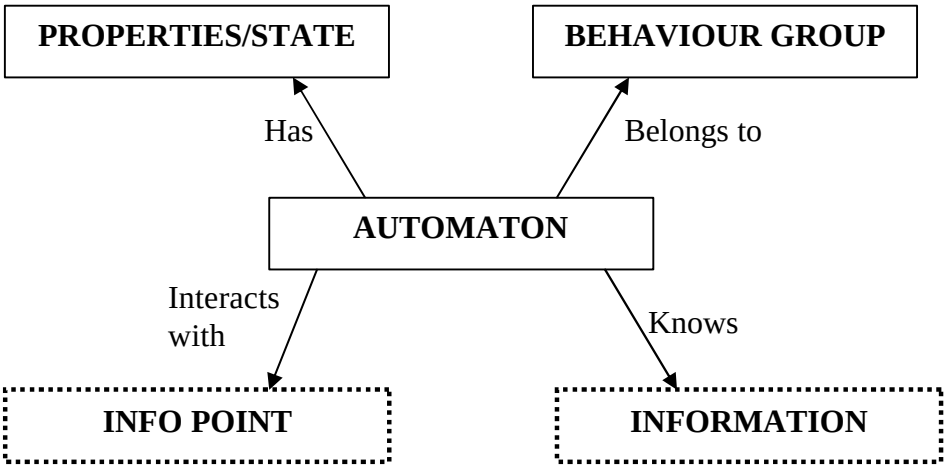
The "information" entity is a drastically reduced representation of a message flowing within the organization. The message is here something abstract and cannot exist other than within an info point or in an automaton. The **information** is part of an **information series**, which denotes that several informations belongs together to form a larger whole. The information series belongs to an **information class**, which denotes a subject the information concerns.

The basic information has very few properties. In essence it is just a serial number with a **priority** and of a certain **medium** (if it is text, sound...). The information can be combined with other informations to **aggregate** a new information. This is an irreversible transformation.

6 Automaton Entity

The automaton entity is rather complex, and is therefore described in a number of different diagrams rather than as a complex whole.

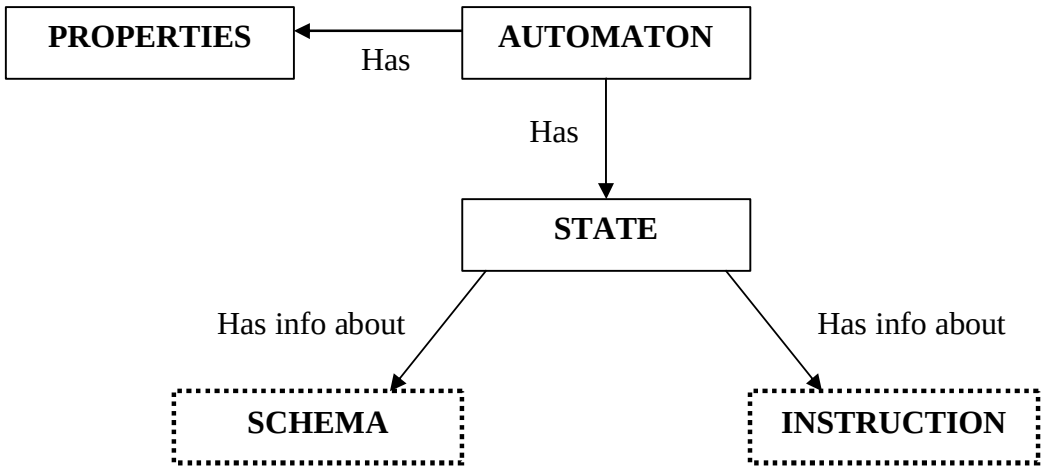
6.1 Automaton Overview



The automaton is an aggregate of a number of properties, and it belongs to a behaviour group, which determines its behaviour. The properties and the state are in the form of parameters where the properties are constants and the state is variable.

The automaton interacts with information points (and with other automatons). It has a list of "known" information. Each time it interacts with an automaton or with an information point, this list is compared with what the point/automaton offers. If unknown but wanted information is offered, an information exchange takes place (if the priority settings allow it).

6.2 Automaton Properties/State



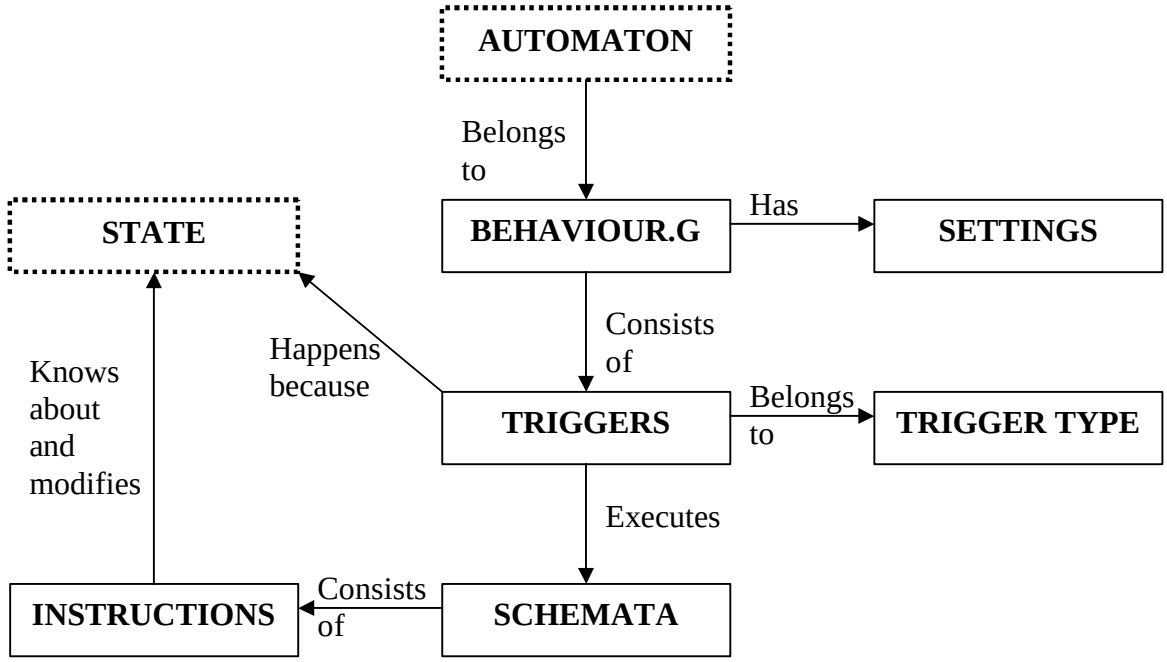
The properties of an automaton are those parameters that are static and pre-set. While the behaviour can take these into account, they cannot be changed during the simulation run. The modelled properties are:

PROPERTY	DESCRIPTION
ID	A unique serial number used to identify the automaton
Position	The position is the position in an organizational sense ("Commander"...)
Function	Function is the organizational unit to which the automaton belongs
Behaviour.grp	The group of schemata the automaton will execute
Home area	Automaton's office or similar (where it goes unless ordered otherwhere)
Start coord	When the simulation starts, this is where the automaton will be placed
Wanted info	The automaton wants information of this class/serie

The "state" is those parameters of the automaton that change over time. The behaviour is aware and dependent of these parameters, and may change them over time. The modelled state parameters are:

STATE	DESCRIPTION
Food	How hungry the automaton is (100 not at all, 0 starving)
Energy	How tired the automaton is (100 not at all, 0 sleeping)
WC	How needy the automaton is (100 not at all, 0 having an accident)
Breakprio	Which priority the current activity has (determines if activity changes)
Coord	Where the automaton is physically
Schema	The current overall activity of the automaton ("going for lunch"...)
Instruction	The low-level activity of the automaton ("Moving to area 24"...)

6.3 Behaviour



The foundation of the behaviour in the simulation is the trigger-schema-instruction triad. Most of the behaviour is implementation-specific and therefore not modelled. Only the low-level instructions and the trigger types are specified in the model. For the rest of the behaviour, the parts merely specify how these can be combined and connected. The trigger types specify both triggers possible to include in a group description, and state triggers.

The modelled trigger types are:

TRIGGER	DESCRIPTION
Timetrigger	Executes a schema at a given time
Infotrigger	Executes a schema when a info collection goal has been reached
Eventtrigger	Executes a schema when something happens in the environment
WCTrigger	[STATE] Executes a schema when WC drops below a certain level
Foodtrigger	[STATE] Executes a schema when Food drops below a certain level
Energytrigger	[STATE] Executes a schema when Energy drops below a certain level

The modelled low-level instructions are:

INSTRUCTION	DESCRIPTION
SetValueAbs	Sets a state parameter or a variable to a explicit value
SetValueRel	Modifies a state parameter or a variable with a relative value
SetSchema	Terminates execution of current schema and starts another
GotoPoint	Go to a specific information point
GotoPointClass	Go to the closest information point of the given class

ContinueSchema	Continue a schema temporarily overridden by a triggered schema
Work	Execute work (collect/distribute info in current room)
Meeting	Execute meeting (go to middle of room and interact with other automata)
GotoArea	Go to a specific room
GotoAreaClass	Go to the closest room of the specified class
Passive	Do nothing (except interact with automata) for a while
SendInfo	Send information to a specific automaton
RecInfo	Try to get information from a specific automaton

7 Implementation of the AMSIDO-1 models

Below is a discussion about the actual implementation of the models, something which as mentioned is not to be confused with the actual models.

7.1 Schemata

Below are a number of implemented schemata presented. Note that these are not a part of the actual simulation program, but are configurable for each case scenario. Only the triggers and instructions (see above) are part of the actual simulation software. The schemata are summarily presented her mainly as an example. Please refer to the XML files of the reference implementation for more details about these schemata.

SCHEMA	**DESCRIPTION**
init	This is the first schema called. It sets state and calls "work" schema.
coffee	Called when energy gets low. Moves to coffee room.
wc	Called when WC gets low. Moves to wc area.
work	Basic work schema. Always called when other tasks are finished.
stdwork	Interact with points in room until interrupted
bigfood	Lunch schema. Moves to lunch room.
smallfood	Same as "coffee" but called at scheduled times.
tv	View TV broadcast. (move to closest TV)
overwork	Continue work (meant to be called when relieved) [not finished]
sendinfo	Send known info to correct automaton [not finished]
order	Accept orders [not finished]
meeting	Have a meeting (go to meeting room and execute MEETING)
bigstaff_meeting	Have a meeting with larger group [same as meeting]
form_meeting	Have a more formal meeting [same as meeting]
cc_meeting	Specific target group (land, air..) meeting [same as meeting]
smallstaff_meeting	Staff coordination meeting [same as meeting]
decision_meeting	Choice of alternatives meeting [same as meeting]

7.2 Behaviour Groups

As with the schemata, the behaviour groups are not really a part of the simulation software, but something that is modelled outside the actual simulator. These are the behaviour groups in the reference implementation (again, for more information about these, please refer to the XML files).

GROUP	DESCRIPTION
planning_function	Members of the J*-functions, coordinating the effort
realization_lma	Target group members implementing decisions
own_function	J*-function members working in specific staff
own_lma	Target group members working in specific staff
control	Command and control function
planning_lma	Target group member coordinating the effort
realization_function	J*-function member implementing decisions

7.3 Events

The "Events" is currently a convenience function for inserting new information into the simulation. The events is merely a list of which information packets to insert into which information point at which time. The event can be either INFOEVENT which inserts a specific information packet, or SERIALEVENT, which starts inserting packets from a certain series at a given interval.

7.4 Navigation

The current automaton navigation is built with a brute-force algorithm. Before the simulation starts, the routes between all possible locations are calculated. These routes are stored in a large vector, which the navigation module consults whenever an automaton is about to make a decision on in which direction to move.

The vector is cached in data file which is read the next time the simulation starts. For the reference implementation, the file consists of approximately one megabyte of XML code, and takes five minutes to calculate.

The reason for the use of this approach was mainly efficiency. The navigation is the most processor-intensive part of the simulation, and the simulation would run significantly slower if the routes had not been precalculated.

However, the navigation module is easily replaced as the simulation communicates with it through a limited interface and does not care how the actual responses to the direction queries are produced.

7.5 Unimplemented Model Parts

Below is a brief summary of the major differences between the models and the actual implementation. Please also refer to the "lowlights" section in the end report of AMSIDO-1.

7.5.1 Arena

The arena is mostly implemented, but only for one floor. It is currently not possible to have several floors in the organization, and thus the stair object is not implemented.

7.5.2 Information Points

The information points exist as such, and they receive and offer information to automatons. Thus the basic functionality is implemented. However, while all properties exist in the object, they are currently mostly ignored. Thus, it is perfectly possible to send a video message into a telephone or hang sound on a bulletin board. With these deficiencies, all information points are to be considered as being exactly like each other, something which is quite far from the underlying model.

7.5.3 Information

All the modelled parts of information exist in the information object. However, as medium and priority is never taken into account by the other objects (automata and information points), it can hardly be said that they are implemented. As such, information can currently not be considered more than a serial number.

7.5.4 Automata and Behaviour

The automatons as such are implemented as modelled, but the behaviour lacks several key features:

- There is no regard taken to the properties of offered information and thus all automata want all information.
- The automata do not *do* anything with the information they find, they just collect it. This is, however, probably an issue also with the underlying model.
- The instructions SENDINFO and RECINFO are not implemented.
- The instruction WORK currently moves the automaton around in the room to visit all points in the hope that new information has arrived. This is not a good implementation of work behaviour.
- There is currently only one type of meeting.
- INFOTRIGGER and EVENTTRIGGER are unimplemented.

8 AMSIDO-1 Model Deficiencies

Below is a summary of the things found lacking in the models before a more formal evaluation was started (the results of the formal evaluation can be found in AMSIDO-2, Validation and data collection). While the section is called "Model deficiencies", not all of the below proposals are sensible to implement. They should be considered as pointers to areas that should be investigated further.

8.1 Arena

The arena is the most concrete part of the models, and is as such probably not something that will develop significantly over time. However, some issues can be found here.

Firstly, the simulation projects claim to simulate information within "geographically delimited areas". In spite of this being quite general - it could as an example fit a small forest area - the current model has a distinct house/building bias. Ideally, the arena should be able to be what is claimed in the project goals: any geographically delimited area.

Secondly, the arena is entirely without properties currently. This in spite of the fact that it is reasonable to believe that environmental properties such as light, ventilation and similar have noticeable effects on information distribution. As an example, it is reasonable to think that it is difficult to read messages on a bulletin board in a completely dark room.

8.2 Information Points

The information points are, as modelled, very limited. Firstly, it is assumed that all information points are immediately comparable, while in reality few similarities exist between a radio and a bulletin board. Possibly, the information points needs to be categorized in a few groups and the parameters be spread between these based on their generality. It would likely be easier to find additional and more relevant properties describing the information points if such a categorization was done.

In the current models the information points do only have "properties", which might be an unnecessary reduction. It is conceivable that the points could also have "behaviour" and take some limited initiatives. As an example an email station could say "ding!" when it gets an email in order to get attention. The same goes for the points' "state": as an example some points should be possible to "switch off" which would cause them to be temporarily unreachable. Another example is that they might break and need repair.

Further the possibilities for interaction are somewhat limited currently. An information point is a static thing which accepts or offers information. As an example to further interaction, the automata might be able to configure parts of the point, or even move it around.

Lastly the information points are considered as monolithic wholes, something which is not really a good description of, as an example, a computer. A computer behaves quite differently depending on whether email or web is used.

Most of the above issues were considered and theoretically handled in an MSc thesis written within the frame of the AMSIDO-2 project during the spring (see Information node characteristics for model construction.).

8.3 Information

The information was initially handled as a kind of "token" without any intrinsic properties. It was thought that it would be possible to handle information this way, as only the routes for its distribution was interesting - not its contents.

Unfortunately this led to a very fuzzy definition of what information is, and this affected the behaviour of the automata in unforeseen ways. As an example it became difficult for the automata to know which information to look for. Another example was that the token approach led to the fact that all information has to be known to the modeller in advance, which is both impractical and unrealistic.

So the main issue with "information" is that it needs a definition that clearly takes into account its contents, and which allows information to be classified, handled and generated. The definition (and new model) needs to take into account that "information" is more like "message", which can contain things like orders, instructions and requests.

With a better model/definition of information, it should ideally be possible to replace or improve the aggregate model which was neither practical nor representative.

Another issue with the information is that it currently (and probably also in the future) only describes explicit "messages". Implicit information like body language or moods is not transmitted between automata. It is, however, not likely that things like this could be efficiently modelled.

Most of the above issues, and several issues that are not listed above, were considered and theoretically handled in an MSc thesis written within the frame of the AMSIDO-2 project during the spring (see Making the concept of information operational).

8.4 Automata and Behaviour

8.4.1 Properties

The automata are currently not described at all as individuals. An automaton is merely a part of a group with preferences or individual properties. Ideally the automaton should be able to act as individuals, or in other words have personality, feelings and similar. As an example persons with different personalities might not work well together.

Somehow these parameters should either be possible to configure per automaton, or they should be randomized. For randomization, there should be some sort of normal distribution for how the properties vary across the sample.

8.4.2 Information

Mainly depending on the unclear definition of information, it is unclear where information should go and why. While the idea that it is the automatons and information points that actually moves the information is basically sound, the actual addressing of the information should probably be moved from the automaton behaviour to be a property of the information.

Further, the relation between the automaton and the information is a bit unclear. As an example there is no representation of whether the automaton "understands" the information, and there is no representation of the context and background for the situation when an automaton receives information.

8.4.3 Behaviour

The basic model with instructions, schemata and triggers seems to hold together as for now, but probable needs to be extended. Both the instructions and the triggers need to be more elaborate. As an example, the instructions need control structures (selections and iterations).

All in all, the model might benefit from moving some of the behaviour from the general simulator to the automatons. Currently, the automatons do not have sensors of their own, and can thus not react to their environment. It is rather the simulation as a whole that determines, as an example, that two automatons are close enough to interact.

The work and meeting behaviour need an overhaul. Work is currently modelled as the production of information aggregates which is not necessarily a good representation of what work is. There is no model of what happens with spare time, or when the automaton is "finished" with a task.

8.4.4 Groups and social phenomena

The AMSIDO-1 models are entirely focused on the single automaton for all behaviour, and social behaviour is considered an aggregate of individual behaviour. This is not entirely practical as groups (meetings, social situations, automaton-automaton interactions...) could be easier described as such rather than to describe everything on an atom level.

8.4.5 Development

All behaviour in the simulation is currently entirely static. The automata have no memory, and do not change or develop over time. Ideally, some development or learning processes in regard to information distribution should be represented.

8.4.6 Other

Currently the time base for the simulation is a single day. It would be nice to be able to run larger simulations to see how things develop over, say, a month.

8.5 Missing entities

The above describes the deficiencies with the *existing* entities. However, it is not necessarily so that the model takes into account all the relevant entities that could be modelled. To speculate, at least three more entities could be conceived as relevant to model.

8.5.1 Environment

Currently the organization hangs in a void. There is no model at all for how information should enter the organization or why. Neither can the organization as such produce an output. Some of this could be addressed by modelling an "environment" object with limited properties and behaviour.

8.5.2 Organization

As mentioned above, the organization is considered an aggregate of automata. However, it might be useful to be able to describe the organization as such and give it properties and behaviour to which the automata could react. As an example, the organization could have policies or determined routines for how things should be conducted.

8.5.3 Non-information artefacts

The only artefacts modelled currently are the information points. These are of course the most relevant for the information distribution, but there are also other artefacts that might influence it. As an example, the coffee machine might not spread information in itself, but it occupies the attention of the automaton for some amount of time during which no information distribution takes place. Perhaps some behaviour or properties of this kind of artefacts should need to be modelled.

9 Conclusions

So far in the project it has not been relevant to study the output of the simulation software, because of three major reasons:

1. The underlying models was only recently evaluated empirically (see AMSIDO-2, Validation and data collection).
2. Even before the evaluation, several model deficiencies were known
3. The models, as far as they exist, have not been fully implemented

Therefore, the *models* have been evaluated more formally.