High-level Petri Nets for a Model of Organizational Decision Making

Sven Heitsch, Michael Köhler, Marcel Martens, and Daniel Moldt

University of Hamburg, Department of Computer Science
Vogt-Kölln-Str. 30, D-22527 Hamburg
{5heitsch, koehler, 5martens, moldt}@informatik.uni-hamburg.de

Abstract. This paper introduces a new direction in formalizing sociological models. Sociological theories are a field of application for computer science, hence sociologists describe a theory in informal ways. These theories are transformed into computational models which can be studied and investigated with formal methods. We have chosen to formalize a common model of sociological theory, the Garbage Can Model of Organizational Choice. This model refers to organizations as organized anarchies. Bounded rationality leads to ambiguous decision situations.
Colored Petri nets offer formal semantics, graphical representation, means to model concurrency, and immediate executability and, thus, seem to be well suited to present complex sociological dependencies. Our approach uses a special formalism of high-level Petri nets, called Reference nets, which is applicable to present the individual parts of the model.
By the executable Petri net model of the Garbage Can theory, central notions like concurrency, conflict, and confusion – known in the Petri net theory – could be directly expressed and presented to sociologists and lead to new insights to some underlying concepts of this sociological theory.

Keywords: Garbage Can Model of Organizational Choice, Net Instances, Object-oriented modeling, Petri nets, Reference Nets, Synchronous Channels

1 Introduction

"Socionics" as understood in this context stands for an interdisciplinary research field between sociology and computer science. Research aims at the question how models of the social environment can contribute to the development of intelligent computer technologies. The modern society delivers a rich reservoir of ideas for the modeling of multi agent systems. Vice versa sociology may benefit from computer science by using computer science techniques for the validation and discussion of their terms, models, and theories. Finally future applications of "hybrid communities" are a special challenge to both disciplines (see [Mal99]).

Finding a formal description for sociological theories is a problem. Sociologists formulate their theories usually only in a textual form. This does not allow the application of more formal methods to validate their theories. One of the main ideas presented here is to formalize the theories by building Petri net models. A model which can be executed by computers and validated by sociologists is desirable. This paper focuses on the sociological theory of the Garbage Can Model of Organization Choice by Cohen, March, and Olsen ([CMO72]). Here a Fortran program was used, later Masuch and LaPotin in [ML89] presented an implementation with a functional language. The main drawback is that these models can be interpreted by computers, but not by many sociologists. The Garbage Can Petri net model is a first step to develop a special Petri net formalism which allows sociologists to directly identify their concepts.

Besides notions of multi agent systems from Distributed Artificial Intelligence Petri net modeling seems to be a promising approach. Object-oriented concepts in the context of programming languages (e. g. [Lou93]) and the Unified Modeling Language (UML) ([JBR99]) combined with high-level Petri nets are the techniques used in this paper.

Petri nets provide concurrency, conflicts, confusions, active and passive views (see [Pet81], [Rei92]). Sociologists use similar terms to describe their models. Object-orientation is used for finding the overall structure because objects represent real-world phenomena. [Val96] introduced the notion of "nets in nets", [Mol96] presented a general approach using Object-oriented Colored Petri Nets. Adding net references and integrating concepts of object-oriented programming resulted in a high-level Petri net formalism called Reference Nets (see [Kum99]). Theory was put into practice by the implementation of Renew, the Reference Net Workshop (see [KW99]).

In this paper the application of Reference Nets to the Garbage Can Model of Organization Choice ([CMO72]) is featured. [CMO72] is a fundamental paradigm of behaviouristic organizational theory. The article has been and still is widely received among sociologists. An illustrative, but semantically precise model of the sociological theory which can easily be simulated is the socionical contribution of this work. An early version of the work in progress was presented in [HMM99].

In the scope of the socionics project at the University of Hamburg different formal approaches to sociology need to be considered. This paper focuses on the application of one selected formalism to one sociological theory. In section 2 the interconnection between Petri nets and object-orientation is described. Section 3 gives an insight to the Garbage Can Model of Organizational Choice of Cohen, March, and Olsen ([CMO72]). A reference net model for a generalized version of the theory is introduced in section 4. Then this version is extended to a larger model which captures all aspects of [CMO72] in section 5. Section 6 discusses the Petri net approach, concludes by evaluating the results, and provides ideas for future work.

2 Basic Notions

The basic concepts of object-orientation, Petri nets, their integration, and object nets, that are relevant for this paper are shortly introduced in subsection 2.1. Reference nets, which are used as the modeling technique in this paper, are sketched in subsection 2.2. Some workflow notions, relevant for the basic structure of the incrementally build up modell follow in subsection 2.3.

2.1 Object-Orientation and Petri Nets

Object-oriented analysis is the here chosen method for transforming sociological theories in form of texts into a formal description. Petri nets are used as the formal language. To bridge the gap between object-oriented methodology and Petri nets as description technique, concepts of the former are adopted in the latter.

Object-oriented analysis The notions of objects and classes of this paper follow closely the ideas of Louden in [Lou93]. Diagrams follow UML (see [JBR99]). In sociology it is not common to structure models or even theories in that way. Understanding the main ideas of the sociological model and capturing the notions in terms of objects and classes is one aim of this work. The structure of the Petri net models follows the ideas of encapsulation and partitioning common to object-orientation.

Petri Nets A detailed description of the historical development – given by Jensen – can be found in [Jen92]. The basic concepts are concurrency and conflicts, active and passive parts, and the movement of tokens. It is important to provide a visual technique to sociologists since there is a common rejection of formal notions. The few concepts of active (transitions) and passive (places) parts of a system with the restricted relation between them is straightforward and easy to understand.

A first Petri net version of the Garbage Can Model has been developed by Valk using Place / Transition nets. It has been used in lectures of a sociology course at the sociology institute of the University of Hamburg. Sociology students had few problems to understand the model. It even turned out that the Petri net was faster to understand than a natural text, which was taught in previous years.

However, due to the very basic constructs of elementary Petri nets larger models become difficult to handle. This general phenomenon is not specific to sociology. Many high-level dialects of Petri nets have been developed in the last years.

Synchronous communication On the one hand objects can communicate asynchronously by message passing, on the other hand they can communicate synchronously. For example, in programming languages function calls can be used for synchronization and communication. Christensen and Hansen combined this mechanism in [CH94] with Petri nets by introducing typed communication through synchronous channels for Petri nets. Synchronous channels allow different transitions to be synchronized and exchange data.

Object-Oriented Petri Nets (OOPN) Besides the concepts of object-orientation (like object, class, inheritance, and association) the partitioning of large models into special parts has been tackled by computer science. Various approaches how to combine object-orientation and Petri nets have been made (see the work of Lakos in [Lak95], Sibertin-Blanc in [SB94] and Moldt in [Mol96]). Here the idea that an object is a net is followed. The net as an object has a specific interface which allows access to the methods and has an unique identity. Similar objects can be folded to classes. Thus, object nets can be seen as instances of net classes (or templates).

An object-oriented Petri net is a Petri net with a special net structure. For a more compact notation usually Coloured Petri nets are used (see [Jen92]). The structure ensures that a certain kind of interface of the object is implemented. Each method is represented by a transition (with some kind of inscription), each variable can be represented by a place with a certain color set. By the appropriate marking the state of the variable (and of the object) is represented. The identity is ensured by the net structure itself. When Colored Petri nets are used, these ordinary nets can be folded and then the color determines the right assignment of each net element. The tokens are related by appropriate kinds of tuples.

Object Petri Nets (OPN) As objects are instances of a certain net class each object has an unique identity. This identity can be referenced by other objects an arbitrary number of times. Thus, the reference to a net is the token of another net. Due to the reference semantics[1] the nets are objects which are located somewhere and the tokens are only references, as opposed to the value semantics, where the nets themselves reside on a place.

This leads to the question, whether nets could be regarded as tokens and to the topic of self reflexivity in the Petri net theory. The instrument of "nets as tokens in a net" of [Val98] defines system nets which provide the environment for object-nets to move and communicate. This idea is illustrated in Fig. 1, where one can see a system net having an object net as its token. The object net behaves like a token, so if the transition t_1 fires it removes the object net from place s_1 and outputs two nets – one in place s_2 and one in s_3. This is quite different from the meaning of reference semantics – illustrated in Fig. 2 – where two references to one single object net would have been placed in s_2 and s_3.

As discussed in different works (cf. [Val96] and [Köh99,Köh00]), we obtain problems, if we regard these tokens as a whole net with its own marking (as in Fig. 1) and also, if we regard these tokens as references to the original net (as in Fig. 2).

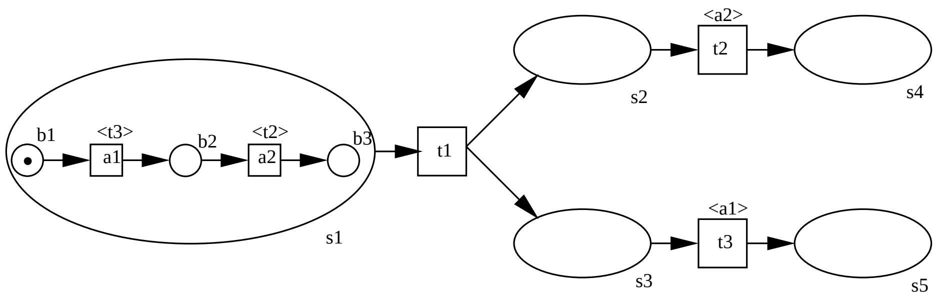

Fig. 1. An object net as a marking of the system net

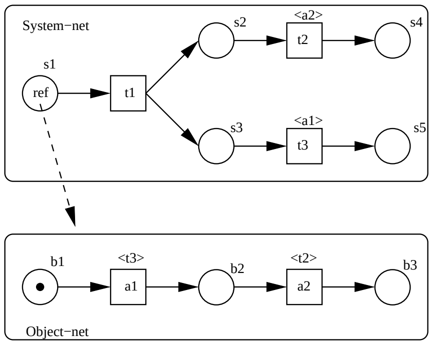

Fig. 2. An object net as a reference token

[1] Reference and value semantics for Petri nets should be understood analogous to the concepts in programming languages.

We use a conceptional notation – besides the concrete syntax of the Renew conventions – for synchronous channels. A transition t could be labeled with another transition, like $\langle a, b\rangle$. This means, that the transition t must fire synchronously either with transition a or with transition b. In our example the transition t_2 is labeled with $\langle a_2\rangle$, so t_2 and a_2 must fire synchronously (see Fig. 1 and Fig. 2).

Have a look at figure 2, where we have one object-net and one system net. In figure 3 one can see the a firing sequence, which is possible under the assumption of reference semantics.

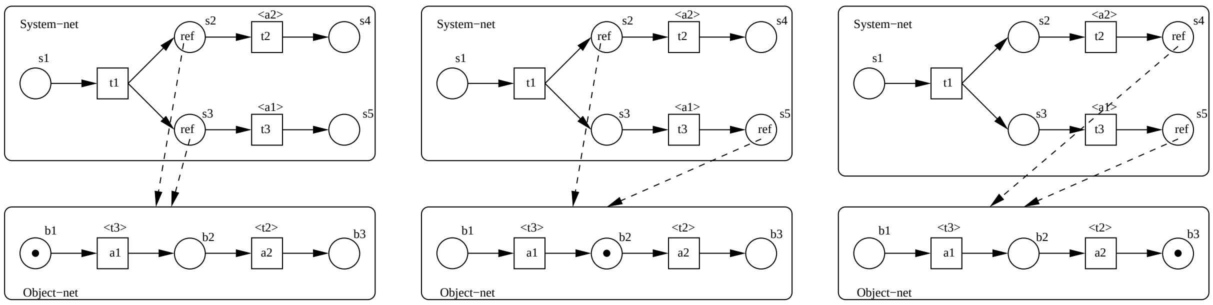

Fig. 3. Snapshots, reference semantics

You see, that our intuition to have two independent copies of our object net is defeated. Instead, firing of a transition changes the marking of a place somewhere else in the net.

If we try to fire this sequence again with a value-oriented semantics we run into trouble, as one can see in figure 4. In the situation of the right net in figure 4 no transition is activated, since the two copies of the object net are unrelated in their markings.

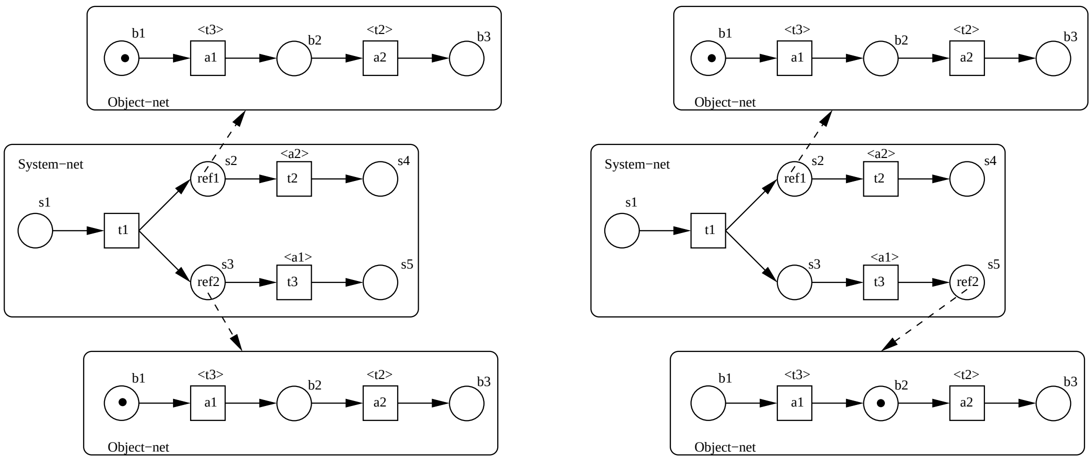

Fig. 4. Snapshots, value semantics

On the other hand, one can show, that a value oriented semantics – in such a simple way, as presented here – also has its disadvantages. To overcome these problems, Valk has introduced a “process semantics” for markings – an idea, which is not going to be discussed here.

The benefit to use references to nets is efficiency of implementation and the well known behavior as used in usual programming languages. From a formal point of view, the reference semantics destroys the Petri net paradigm of locality. As we have decided to use the reference semantics for our implementation of the garbage can problem, our design has to take care of the loss of the locality argument. The solution here is to use only one reference to a net to overcome this design problem. This is a special case of the approach in [Köh00]. In this case value and reference semantics coincide.

2.2 Reference Nets

In this paper the formalism of reference nets is used which incorporates the concept of Valk in [Val98]. Reference nets – as implemented by the Renew tool – are a special high-level Petri net formalism that provide dynamic creation of net instances, references to other net references as tokens, and communication via synchronous channels ([Kum98]). Java is used as the inscription language.

Basic elements Reference Nets (as Petri nets) consist of three types of elements: places, transitions, and arcs. Semantic inscriptions can be added to each net element. Places can have a place type and an arbitrary number of initialization expressions. On creation of a net instance the initialization expression is evaluated and leads to the initial marking of the net. Arcs can have arc inscriptions. The arc inscriptions are evaluated when a transition fires and the results determine the consumption and creation of tokens. Transitions may carry diverse inscriptions. There are expression inscriptions which are performed when a transition fires. Guard inscriptions are preconditions to the transitions, i.e. a transition is only activated if all attached guard expressions evaluate to true. Action inscriptions start with the keyword action and are only evaluated when the transition fires. Creation inscriptions (consisting of a variable name, a colon, the reserved word `new` and the name of a class net) create new instances of nets.

Synchronous Channels Synchronous channels synchronize two transitions which both fire atomically at the same time. Both transitions must agree on the name of the channel and on a set of parameters before they can synchronize. This concept is generalized by allowing transitions in different net instances to synchronize.

The initiating transition must have a special inscription, a so-called downlink, which makes a request at a designated subordinate net. The syntax for a downlink consists of the name of the net reference, a colon (:), the name of the channel, and an optional list of arguments. The requested transition must have an uplink as an inscription which serves requests from every other net instance. Every time a synchronous channel is invoked, the channel expressions on both sides are evaluated and unified. A downlink is specified as a transition inscription `netexpr:channelname(expr, expr,..)`. An uplink is specified as a transition inscription `:channelname(expr, expr,...)`.

This syntax illustrates the semantic difference between uplinks and downlinks because the invoked object net must be known before the actual synchronization can begin. A transition can have an arbitrary number of downlinks, but at most one uplink and a transition with an uplink can not fire without being requested explicitly by another transition with a matching downlink.

Net Instances and Net References When a net is constructed, it is a template without any marking that is used to create an arbitrary number of net instances during the simulation. These instances have got a marking, which can change over time.

A net instance is created by using the reserved word `new`, a variable, and a net name in the following syntax `var:new netname`. This means that a new instance of the template `netname` is created and bound to the variable `var`.

Whenever a simulation is started, new net instances are created by transitions that carry the above mentioned creation inscriptions. For any further access on those new net instances now their references, which are tokens, are used.

The powerful formalism allows to model technical applications as well as business applications, especially workflow systems.

2.3 Workflow concepts

To describe the control flow within net models capturing the organizational structure of a company the workflow nets of van der Aalst [Aal97] can be considered as a kind of standard. Within this contribution organizational aspects are of central interest, however, the main emphasis is not the economical point of view, but the sociological one. Even so the modeling requires to adequately present the control flow. Therefore, the structure of the nets is designed to fulfill the definition of Wil van der Aalst from a structural point of view. The extension to coloured Petri nets and especially reference nets can briefly be described as providing no dynamic violation of the workflow net criteria, even if the structural design with respect to places, transitions, and arcs violates it. This is due to the net inscriptions.

Looking at the examples in the next section will reveal that the control flow could even be designed in a very rigorous way for the Garbage Can model. Nevertheless, the aspects of workflow are not discussed here any further. A more intensive discussion can be found in [MV00] and [AMVW99]. There the relations between object-orientation and nets as tokens and workflows are discussed. Furthermore, an architecture for a workflow engine is presented.In this paper the applicability to a sociological example is challenged. To find a reasonable starting point an example from organizational theory has been chosen.

3 A Garbage Can Model of Organizational Choice

In sociology decisions are seen as one of the main outcome of organizations (see Luhmann in [Luh88]). This section introduces the Garbage Can Model of Organizational Choice by Cohen, March, and Olsen (see [CMO72]). Then a generalized version of the original work is presented. This will be the basis for the executable Reference net model of the following section.

The article of Cohen, March, and Olsen is a fundamental and often cited contribution to behaviouristic organization theory ([Imh99]). The model combines empirical characteristics, theory, and simulation aspects. It also deals with the essential sociological task how organizations can survive in an ambiguous and complex environment. The Garbage Can Model represents a criticism to common rational choice theories because decision making is seen as an ambiguous situation. It is argued that the behavior of at least parts of any organization can be described with this model.

The Garbage Can Model considers organizations as organized anarchies where decision situations are characterized by three general properties: problematic preferences, unclear technology, and fluid participation. It is argued that a decision is the outcome or interpretation of several relatively independent streams within an organization:

– A stream of problems: Problems are determined by inner and outer organizational circumstances and require attention of participants. Problems are looking for situations in which they might be raised.
– A stream of energy from participants: Participants come and go. It is assumed that they provide energy for organizational decision making.
– A stream of solutions: Participants of the organization produce solutions. Solutions move around, actively looking for questions to which they might provide an answer.
– A stream of choices: Choice opportunities represent the point of time when a decision is required by the organization. Each choice opportunity can be seen as a garbage can into which diverse problems and participants are dumped.

Organizations can be viewed as collections of choices, problems, and participants. Participants and problems migrate between the different garbage cans. If a participant meets a choice under the right circumstances, a decision can be made. If there is at least one problem attached to the choice, the making of a decision leads to a rational outcome (decision by resolution), the problem is solved. Or the making of a decision takes too long and no problems are solved (decision by flight). If the decision is made so quickly that no problem has the chance to come up, it was made by oversight (decision by oversight).

Masuch and LaPotin provide in [ML89] a metaphorical view on the basic Garbage Can processes of decision making:

> "... reconsider the finale of the James Bond movie 'A view to kill'. Agent 007 balances on the main cable of the Golden Gate Bridge, a woman in distress clinging to his arm, a blimp approaching for rescue. In terms of the Garbage Can Model, the blimp is a solution, Agent 007 a choice opportunity, and the woman a problem. In the picture's happy ending, the hero is finally picked up, together with the woman, and a solution by resolution takes place; the problem is solved. Now imagine numerous blimps, women, and heroes, all arriving out of the blue in random sequence. Heroes take their positions on the main cable. Women cling to heroes, blimps hover above the scene. Heroes may or may not be able to hold an unlimited number of women, but the blimps carrying capacity is limited; heroes with too many women cannot be rescued. Blimps are retrieving rescuable, i.e., not-too-heavy, heroes. Women in distress are aware of that and switch heroes opportunistically, choosing the hero closest to retrieval. As women, as well as blimps, make their choices independently of each other, a light hero, on the verge of rescue, may suddenly find himself overburdened. Heavy heroes, in turn, may become rescuable all of a sudden as their women desert them."

This coming and going is the mechanism called fluid participation. Women may not be saved at all if they change between heroes disadvantageously and all of their heroes of choice turn out to be too heavy; then, these problems are not solved. Heroes may be saved when all women just have left; this is called a decision by flight. Also, heroes can be rescued before any distressed women was able to hold on to him; then, a decision by oversight has occurred.

Let's come back to the grounds of organizational theory and sum up the terminology: the bridge is an organization, heroes are choices, women are problems, and blimps are solutions. Choices attract problems and solutions. A choice is made if there is an appropriate solution to its problems. Three styles of decision making may appear, but only one of them solves problems.

One might wonder where the participants have gone. In this rather generalized version by Masuch and LaPotin in [ML89] participants do not appear. They remain backstage and have an indirect impact on the organization. They produce solutions and throw them into the scene. Because the participants are mentioned explicitly in the original Garbage Can Model by [CMO72], they will appear instead of solutions in the following Petri Net models for a better understanding of the decision making process.

4 The Executable Reference Net Model

The results of the above mentioned prototyping approach are three different Petri net versions of the Garbage Can Model.

1. The first version by Valk, regarding the metaphorical view of Masuch and LaPotin ([ML89]), is a non-executable Place/Transition-net because it uses a special kind of arc which allows to remove all tokens of place atomically without knowing the exact number of tokens. This Petri net version is very generalized and reduced. It has been developed to give an impression how to apply Petri nets to sociological theories. It visualizes the essentials of the Garbage Can decision making processes and inspired the following models.
2. The second one is an executable reference Petri net which was created by folding the Place/Transition-net of Valk in order to overcome the constrains of the first version. Colored Petri nets as a higher Petri net level are used. Still, this is a generalized version which illustrates the overall behavior of the Garbage Can model abstracting from details.
3. The third version is an executable reference net model (see [HM99]) consisting of eight large collaborating nets which captures all aspects of the original Garbage Can Model by [CMO72]. It deals with several modeling problems which occured during the prototyping process. Presenting the whole model would certainly be far out of scope in this context.

Since the second version of the Garbage Can Model is presented to full extend in this paper, there will be two nets (Organization and Choice) picked out exemplarily and compared to the complex third version in order to discuss the prototyping approach and the modeling experience. In the following chapter it will be described, how structures of the nets could be reused in more complex models.

In this section the second version of the reference net model is presented. The results of an object-oriented analysis are transformed into object nets. The structure and interaction of these nets are described. Characteristics of the presented model are discussed.

An object-oriented analysis of the generalized Garbage Can Model – as described in the above-cited text by Masuch and LaPotin ([ML89]), but translated into the terms of the original model – leads to the identification of classes and associations (Fig. 5). In terms of Valk (in [Val98]), where Petri nets are used as token objects of other nets, a net which provides the environment and the control for the others is called system net. Tokens of the system net are simply named object nets. In this context there are one system net and three object nets. The methods of the objects are already listed in the class diagram, however, they are explained later in the context of the single classes.

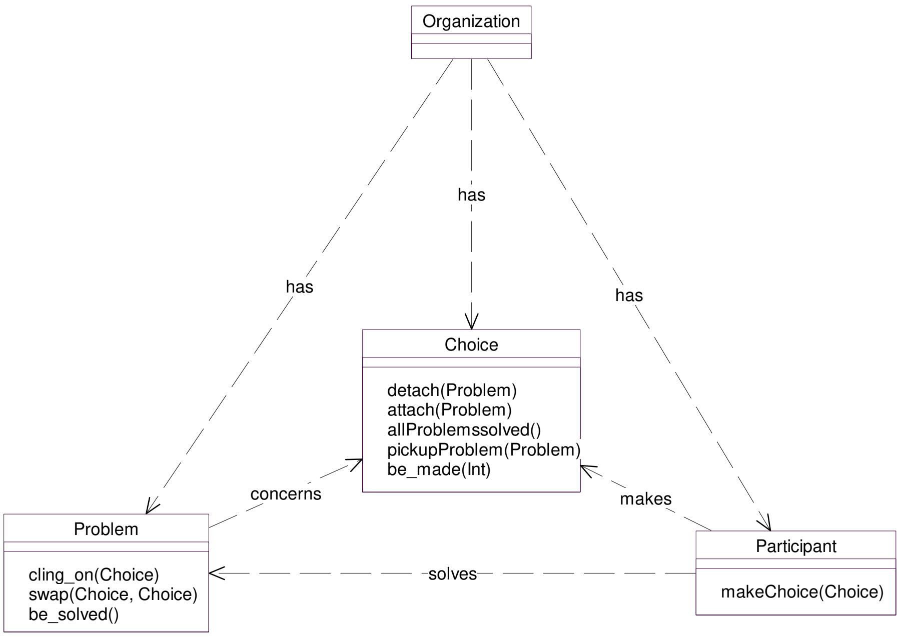

Fig. 5. Class diagram in UML notation

The Garbage Can Reference Net consists of five net classes:

1. the *Organization* which keeps track of the net instances involved and which controls the interactions among those instances,
2. the *Choices* which are the crucial elements of the decision making process and which represent the link between problems and participants,
3. the *Storages*, which are a substructure of the choices supporting the administration of the problems
4. the *Problems* which attach themselves to choices and are solved eventually, and
5. the *Participants* which bring relief to the distressed situation and lead to decision making.

The storage is intentionally not mentioned in the class diagram because it is just an auxiliary net for the choice and has nothing to do with the Garbage Can model.

There will be one instance of the net class Organization which is the system net (Fig. 6). The organization is responsible for instantiating and controlling all other objects involved.

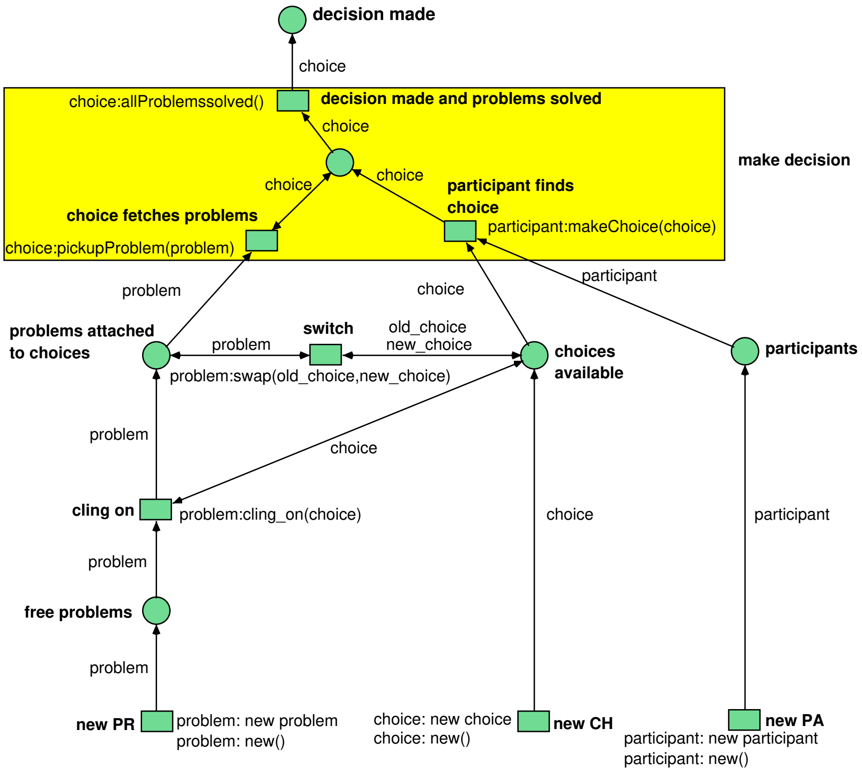

Fig. 6. System net Organization

Choices, problems, and participants are instantiated by calling their "new"-methods. Each transition of the system net calls methods of referenced object nets.

For example, when the transition "cling on" of the system net fires, one instance of problem (precisely, one reference to an instance of a class is meant) and one instance of choice are selected, and the problem's method "cling_on(choice)" is called.

The problem is put into the place "problems attached to choices" and the choice is returned to its previous place. Once again in different wording: transition "cling on" of the system net Organization is synchronized with transition "cling_on(choice)" of one instance of object net Problem. At the same time a synchronous channel called "cling_on" is agreed on by Organization and Problem. A reference to a choice is passed through this channel.

The transition "switch" tells the problem to leave its old choice and cling to a new one. One problem and two choices are referenced. The latter are passed to the problem net via the synchronous channel "swap".

The large gray-shaded box in Fig. 6 can be seen as one large transition which represents the making of a decision. Due to characteristics of the Petri net formalism it has to be split into three single transitions ("participant finds choice", "choice fetches problem", and "decision made and problems solved"). When a participant has been generated and a choice is "available", the transition "participant finds choice" is enabled. This deactivates the transitions "switch" and "cling_on" of the system net. After firing the choice moves on in order to solve all the problems attached to it. Eventually, the attached problems are being fetched, i.e. solved. After all problems of the choice have been removed, the decision making process can be completed (transition "decision made and all problems solved").

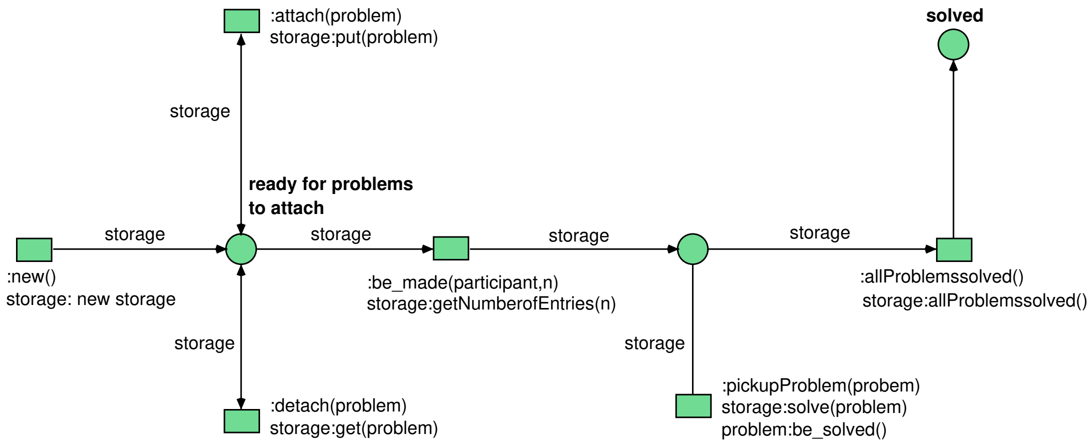

Fig. 7. Object net Choice

Each new instance of the net Choice (Fig. 7) also produces a new instance of the net Storage (Fig. 8) and the reference to this instance is returned and bounded to the variable storage. It is put in the place "ready for problems to attach" at first. Problems may cling to an instance of choice or switch between those. The choice is informed about these changes by the methods "attach" and "detach" and delivers this information to its storage via synchronous channels (put and get), where the number and references of the attached problems are saved. As soon as a decision has been made with the help of a participant (it has received the "be_made(n)" message where "n" represents the number of problems attached to the choice which is passed to the participant through the synchronous channel :be_made) it can neither be left by any of its problems nor can it be clung to by new problems. It is now in the state "decision made" and its "attach" and "detach"-methods cannot be called anymore. The choice's "pickupProblem(problem)"-method is called and returns the attached problems one by one. After all problems have been reported and "solved" by the storage, the transition "allProblemssolved()" is enabled. It fires when the decision making process is complete.

The object net Storage is a special construct to support the Choice with its administration of the attached problems. Basically it serves as a memory for each choice and stores the actual references of the attached problems ("entries") and the number of those references ("number of entries"). Furthermore the storage can move problems to the state entries solved and keeps track of this with a counter of saved problems. By calling the "getNumberofEntries(n)"-method this number "n" of problems is returned. The methods put(problem) and get(problem) add and remove entries and change the number of entries simultaneously. The "solve"-method removes solved problems from the place "entries" and puts them into the place "entries solved" to ensure that all problems are solved before

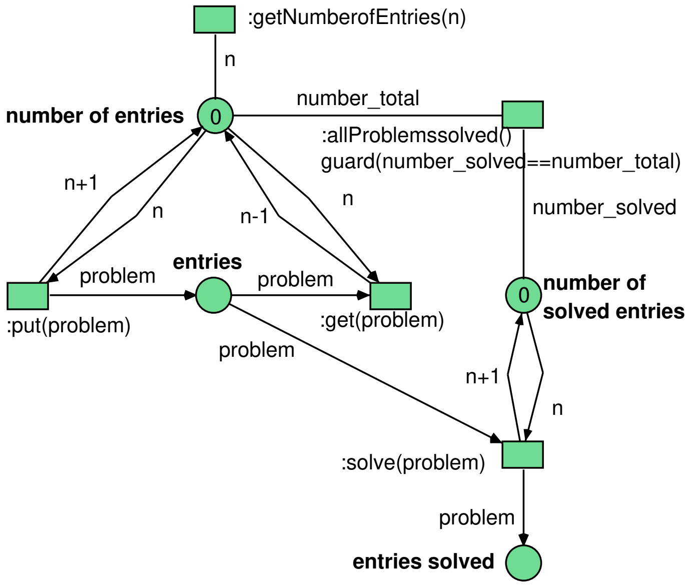

Fig. 8. Object net Storage

the decision process can complete. The transition with uplink ":allProblemssolved()" is enabled, when there are as many problems solved as there are stored.

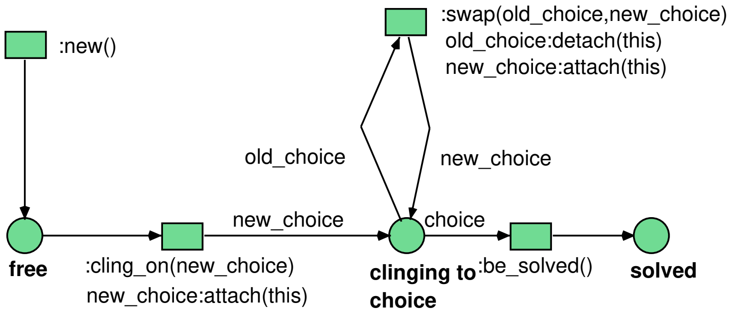

Fig. 9. Object net Problem

After the Problem net (Fig. 9) has been instantiated it will be "free" and looking for a choice at first. Then it can cling to a choice, swap between choices several times, and will finally be solved.

An instance of the Participant net (Fig. 10) approaches the scene and will eventually be relevant to one of the choices. If the choice's number of problems does not exceed the participant's energy (which is an integer constant of 3 in this example), the choice can be made.

The Petri net model puts this version of the Garbage Can Model into a computer executable form. Each object of the sociological theory, except for the storage, can be found and analyzed separately. Also sociologists were satisfied with the visualization and readability of the results.

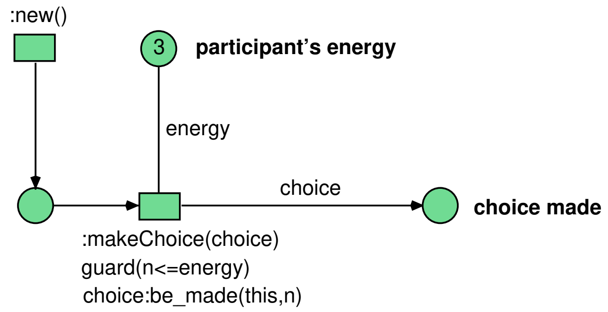

Fig. 10. Object net Participant

The basic phenomena of the original Garbage Can Model appear in this simulation. Especially the three decision styles can be observed. They can easily be visualized by firing sequences of the net during the simulation.

This version covers the illustrative introduction to the Garbage Can Model with heroes, women, and blimps. Intentionally additional features will be treated in the next section.

5 Extensions of the reference net model

In this section the further development of the Garbage Can reference net model (see section 4) will be discussed and illustrated by comparing two nets from the generalized model with the corresponding two nets of the extended model (namely choice and choice (large), organization and organization (large)). Similarities and differences between the net structures will be examined.

The above mentioned reference net model has been extended in [HM99]. The result is a complex reference net model which captures the whole functionality of the Garbage Can Model of Organizational Choice in [CMO72], which is far more complex than the metaphorical approach of Masuch and LaPotin ([ML89]).

The focus during the modeling process was to find basic and reusable structures. This approach was chosen in order to be able to add all the features of the original model without having to change the nets that has been developed so far.

The most important features, which refer to the problems and participants, are the strategy of finding the most attractive choice before clinging on it and the organizational structures which constrain the access to choices (see [CMO72]).

As mentioned in section 3 there is a stream of energy from the participants that is necessary to solve organizational problems. This energy is called "energy available". It characterizes the participants ability of solving problems. The more "energy available" a participant provides, the more skilled he is.

There is an opposite tendency in an organization, which behaves complementary towards the energy available. This is expressed in the "energy required", which characterizes the difficulty of the problems. The more "energy required" a problem has, the harder it is to solve.

Choices attract participants and problems. So a choice receives both types of energy, the "energy available" and the "energy required". A decision can be made if there is at least as much “energy available” as “energy required”. Finding the most attractive choice is done by calculating the difference between those energies: Only if the difference is zero or positive, a decision is made.

One can recognize this behavior in the net structure of Fig. 12 looking at the places which store the energies and the transitions which update and compare them. In [HM99] the afore mentioned organizational structures are also considered. These extensions, strategy, and organizational structures, have been modeled by using additional reference nets (i.e. a net called oracle which has information about all object nets involved and which answers questions, or a net called multiset, which administers available choices, energies and checks the access rights). Since not all details can be presented here, the focus is on the choice and the organization reference nets.

Moving from Fig. 6 to Fig. 11 there is not very much added to the net Organization (large). One can easily recognize the same structure of the net by looking at it. What has been changed is the number of participating objects which is limited following [CMO72], which means that there are ten participants (new PA), ten choices (new CH) and twenty problems (new PR).

Now every instantiation of new objects is reported to the above mentioned object net oracle. Before clinging to a choice or before swapping to another choice problems have to ask the object net oracle for the most attractive choice. Before spending their energy available, participants follow the same strategy as the problems and also ask the oracle for the most attractive choice. There are several more details in this model which can not be explained in the scope of this paper. However, it should be noticed that the oracle is neccessary to implement global knowledge about the available choices and their energy differences.

Participants can spend their energy several times on different choices before they leave the organization by calling the “leave()”-method. The solution process of Fig. 11 is very similar to Fig. 6 if one compares the gray-shaded boxes (“make decision”) and the inscriptions of the transitions involved. When all problems are solved there is a statistic module which analyses the style of decision making.

Comparing Fig. 7 and Fig. 12 again one can point out the same basic structure of the nets. The behavior is basically similar apart from additional places for the energies, the counters and some transition inscriptions.

The choice produces a reference to its storage while being instantiated and remains ready for problems to attach until problems start attaching and detaching. The choice has two places for energies (“energy required” and “energy available”). It can accept energy with the :acceptEnergy(n)-Method. It has an ID in order to be identified and a reference to the oracle. The number of attachments and detachments is stored in the two counters #detached and #attached which is relevant for the statistics at the end of the simulation. Finally, if choices have more energy available than energy required a decision can be made and the storage is emptied gradually (waiting for problems in solution process) until all problems are solved. The detailed interaction with the other nets involved unfortunately can not be explained here.

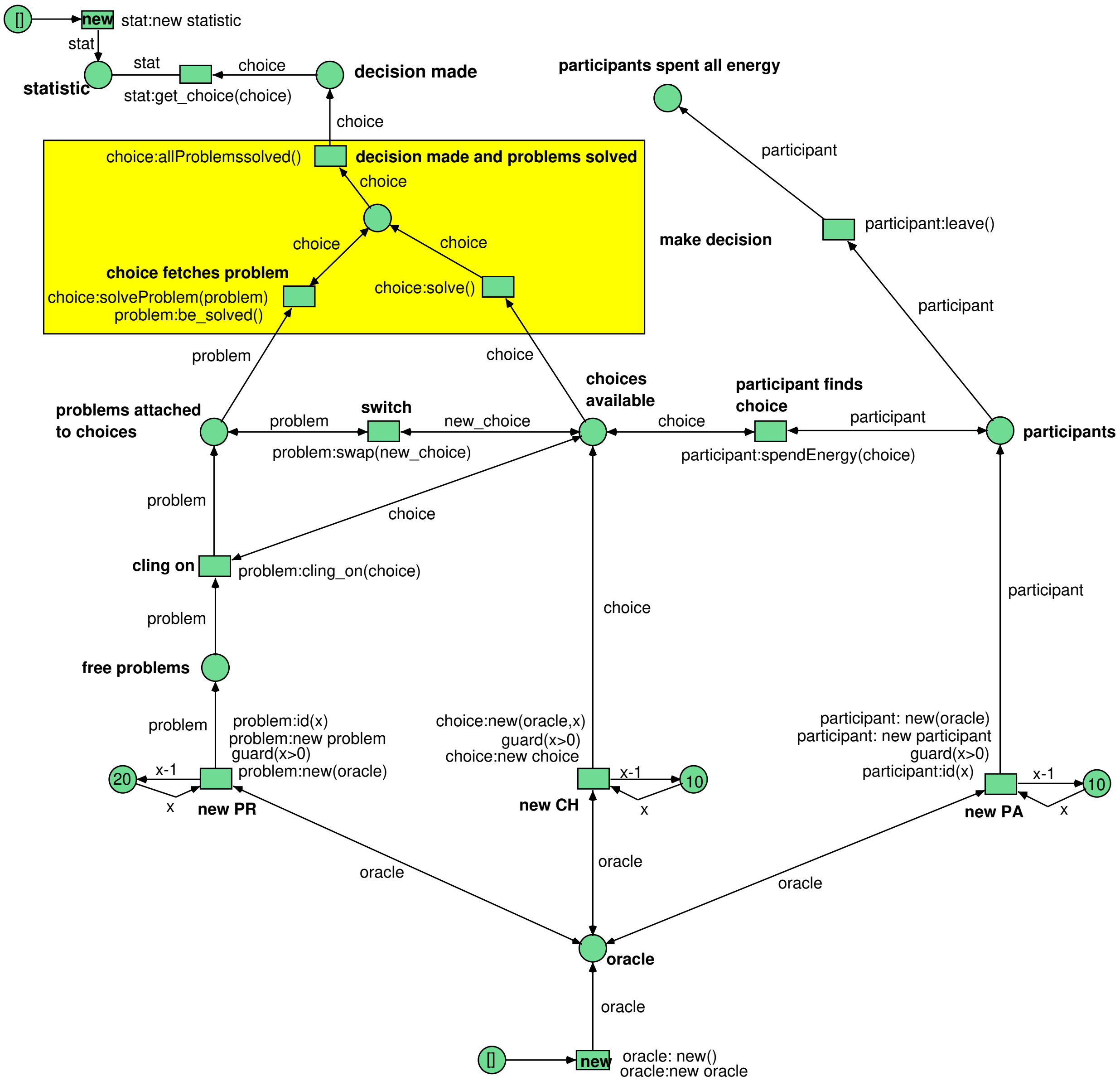

Fig. 11. Object net Organization (large)

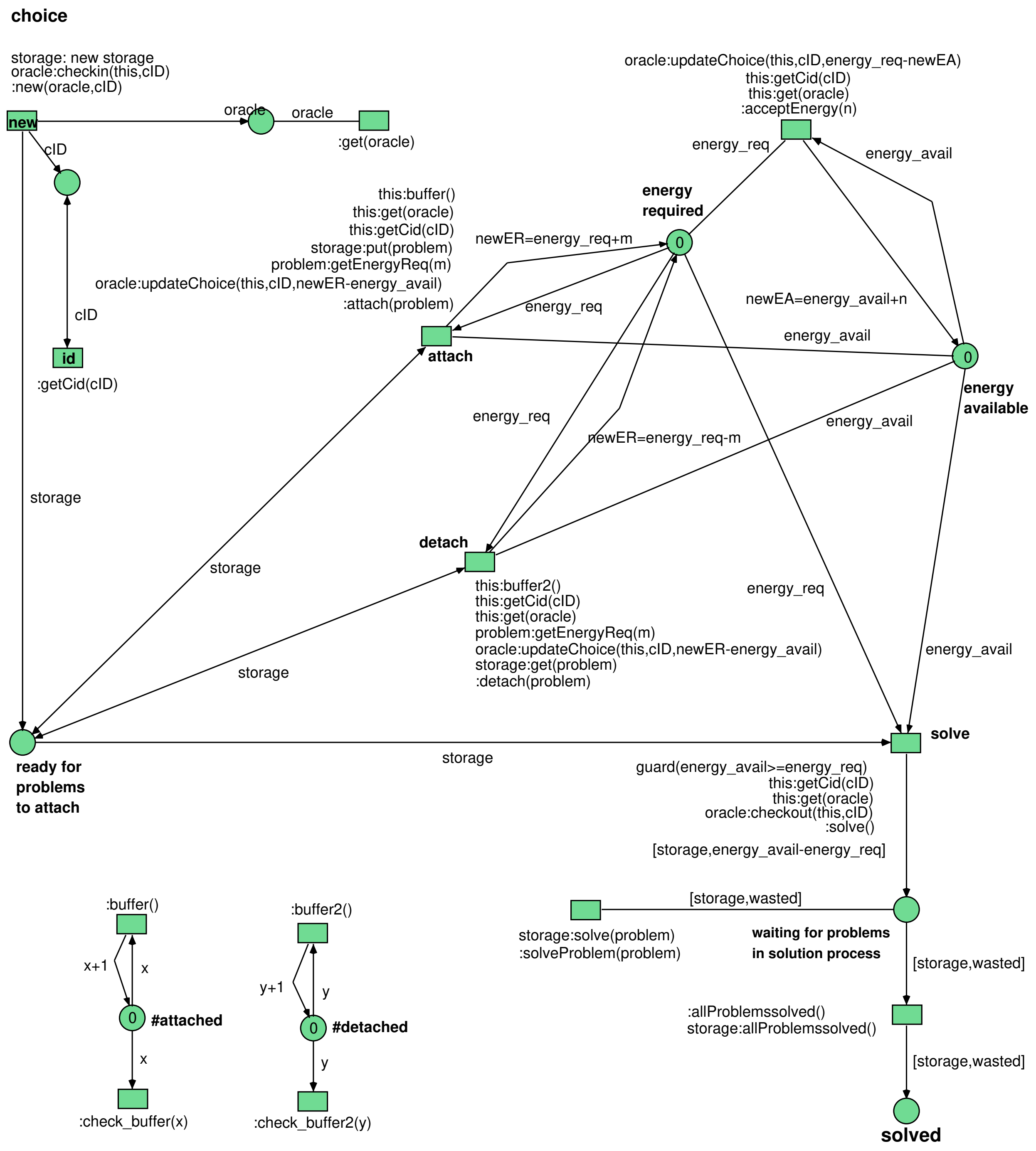

Fig. 12. Object net Choice (large)

6 Conclusion and Outlook

Starting from the problem of formalizing sociological theories a concrete one has been chosen and successfully modeled. By applying Petri nets an operational semantics could be given to the Garbage Can Model of Organizational Choice.

Petri nets and in our case Reference nets have been successfully applied and showed the advantages of its underlying concepts. These are:

- The generally well known advantages of Petri nets, especially the explicit expression of concurrency, conflicts, and confusion.
- The object-orientation to identify objects in the sociological application area, even if advanced concepts like inheritance or polymorphism have not been used.
- The concept of nets as active tokens to separate environment, active elements, and their interaction.
- The synchronous channels and the net instances of the net templates allowed a clear separation of the model and a clear dynamic behavior. Of course, one has to find a compromise between the complexity of the net structure and the quantity of net inscriptions in order to keep the net model understandable.

It should be noted that the success of the project so far crucially depended on the tool Renew, without which the modeling and execution of the model would not have been possible in such a compact way. By treating actors and active parts of the sociological theory as objects for sociologists an intuitive model could be developed.

Resulting from the modeling process and related discussions new questions towards the sociological theory were exposed for and by the sociologists in our group. A detailed discussion can be found in [HHM00]. Some of the results are:

- Many aspects of the sociological model are only implicit and unstructured.
- Global knowledge about available choices is assumed.
- Structural dependencies of the relations between the actors of the model exist.
- Linkage to other sociological theories, which are faded out by the original Garbage Can model, are unreflected.
- Also in the sociological theory situations of concurrency and choice lead to confused actors.
- Different levels of context exist, meaning the sense of environment and restrictions on communication.
- Cooperation of actors has not been taken into account.

Overall, it was interesting to see that the incremental prototyping approach was very successful and that the extension was relatively easy due to the overall object-orientation and the internal organization of the objects in a special kind of scenario nets or workflow nets as described in [Mol96] and [MV00].

The gained improvements and especially the results were inspiring the sociologists in our group to reformulate some parts of the theory. This shows the weaknesses of the traditional view and also allows to extend the Garbage Can theory by some other theories, especially those related to behavioral theories. The structures and processes induced by the additional aspects will lead to better models of organizational choices and will allow our group to apply this new theory to organizational units within public institutions. Especially, we plan to look at decision procedures within universities.

The means to express these new theories shall be developed on the basis of the "nets as active tokens" concept. This should allow for a separate adaptation of the environment and its objects acting within it, capturing the aspects of local changes and of mobility. Furthermore, the concepts of Petri net agents can be developed according to [MW97].

The high-level Petri net formalism could be applied by the computer scientists to a theory of sociology. Within our project we will extend this further to other theories. The results shall be used to enhance agent architectures and shall lead to generalized sociological theories.

7 Acknowledgments

We would like to thank Daniela Hinck and Rolf von Lüde for the excellent sociological input, Rüdiger Valk for the basic model which was the starting point of the Petri net modeling, and Olaf Kummer and Frank Wienberg for their fruitful comments on and support with the Renew tool.

References

[Aal97] Wil van der Aalst. Verification of workflow nets. In Azéma and Balbo [AB97], pages 407–426.

[AB97] Pierre Azéma and Gianfranco Balbo, editors. *Application and Theory of Petri Nets 1997*, number 1248 in Lecture Notes in Computer Science, Berlin Heidelberg New York, 1997. Springer-Verlag.

[AMVW99] Wil van der Aalst, Daniel Moldt, Rüdiger Valk, and Frank Wienberg. Enacting Interorganizational Workflows Using Nets in Nets. In Jörg Becker, Michael zur Mühlen, and Michael Rosemann, editors, *Proceedings of the 1999 Workflow Management Conference Workflow-based Applications, Münster, Nov. 9th 1999*, Working Paper Series of the Department of Information Systems, pages 117–136, University of Münster, Department of Information Systems, Steinfurter Str. 109, 481149 Münster, 1999. Working Paper No. 70.

[CH94] S. Christensen and N.D. Hansen. Coloured Petri nets extended with channels for synchronous communication. In Rober Valette, editor, *Application and Theory of Petri Nets 1994, Proc. of 15th Intern. Conf. Zaragoza, Spain, June 1994*, LNCS, pages 159–178, June 1994.

[CMO72] M.D. Cohen, J.G. March, and J.P. Olsen. A garbage can model of organizational choice. *Administrative Science Quarterly*, 17:1–25, 1972.

[HHM00] Sven Heitsch, Daniela Hinck, and Marcel Martens. A new look into garbage cans - petri nets and organisational choice. In *Proceedings of AISB 2000. Time for AI and Society*, Birmingham, UK, 2000.

[HM99] Sven Heitsch and Marcel Martens. Soziologische Modellbildung mittels Referenznetzen. Studienarbeit, University of Hamburg, Department for Computer Science, Vogt-Kölln Str. 30, 22527 Hamburg, Germany, November 1999.

[HMM99] Sven Heitsch, Marcel Martens, and Daniel Moldt. Petri Nets with Synchronous Channels Applied to a Sociological Example. Work in Progress Presentation at CPN'99 in Aarhus, see URL: `http://www.daimi.au.dk/CPnets/workshop99/`, 1999.

[Imh99] P. Imhof. Social studies of social simulation. URL: http://www.tu-harburg.de/tbg/Deutsch/Mitarbeiterinnen/Peter/Scientific.html, October 1999.

[JBR99] I. Jacobson, G. Booch, and J. Rumbaugh. *The unified software development process: UML; The complete guide to the Unified Process from the original designers.* Addison-Wesley object technology series. Addison-Wesley, Reading, Mass., 1999.

[Jen92] Kurt Jensen. *Coloured Petri Nets: Volume 1; Basic Concepts, Analysis Methods and Practical Use.* EATCS Monographs on Theoretical Computer Science. Springer-Verlag, Berlin Heidelberg New York, 1992.

[Köh99] Michael Köhler. Algebraische Strukturen von Objektnetzen. Master's thesis, Universität Hamburg, 1999.

[Köh00] Michael Köhler. Distribution references and undecided markings. Technical report, University of Hamburg, Dept. for Computer Science. FBI-HH-M-293/00, 2000.

[Kum98] Olaf Kummer. Simulating synchronous channels and net instances. In J. Desel, P. Kemper, E. Kindler, and A. Oberweis, editors, *Forschungsbericht Nr. 694: 5. Workshop Algorithmen und Werkzeuge für Petrinetze*, pages 73–78. Universität Dortmund, Fachbereich Informatik, 1998.

[Kum99] Olaf Kummer. A Petri net view on synchronous channels. *Petri Net Newsletter*, (56):7–11, 1999.

[KW99] Olaf Kummer and Frank Wienberg. Renew homepage. URL: `http://www.renew.de`, University of Hamburg, Department for Computer Science, Vogt-Kölln Str. 30, 22527 Hamburg, Germany, 1999.

[Lak95] C.A. Lakos. From Coloured Petri Nets to Object Petri Nets. In *16th International Conference on the Application and Theory of Petri Nets*, number 935 in Lecture Notes in Computer Science, pages 278–297, Torino, Italy, 1995. Springer.

[Lou93] Kenneth C. Louden. *Programming languages: principles and practice.* PWS-Kent, Boston, 1993.

[Luh88] N. Luhmann. Organisation. In W. Küppers and G. Ortmann, editors, *Mikropolitik. Rationalität, Macht und Spiele in Organisationen*, pages 165–186. WDV, Opladen, 1988.

[Mal99] T. Malsch. Erforschung und Modellierung künstlicher Sozialität. URL: `http://www.tu-harburg.de/tbg/SPP/spp-antrag.html`, August 1999.

[ML89] M. Masuch and P. LaPotin. Beyond garbage cans: An AI model of organizational choice. *Administrative Science Quarterly*, pages 38–67, 1989.

[Mol96] Daniel Moldt. *Höhere Petrinetze als Grundlage für Systemspezifikationen.* Dissertation, University of Hamburg, Department for Computer Science, Vogt-Kölln Str. 30, 22527 Hamburg, Germany, August 1996.

[MV00] Daniel Moldt and Rüdiger Valk. Object-oriented Petri Nets in Business Process Modelling. In Wil van der Aalst, Jörg Desel, and Andreas Oberweis, editors, *Business Process Management: Models, Techniques, and Empirical Studies*, number 1806 in Lecture Notes in Computer Science, pages 254–273, Berlin, Heidelberg, New York, 2000. Springer-Verlag.

[MW97] Daniel Moldt and Frank Wienberg. Multi-agent-systems based on coloured Petri nets. In Azéma and Balbo [AB97], pages 82–101.

[Pet81] J.L. Peterson. *Petri Net Theory and the Modeling of Systems.* Prentice Hall Inc., Englewood Cliffs NJ, 1981.

[Rei92] Wolfgang Reisig. *A Primer in Petri Net Design.* Springer Compass International. Springer-Verlag, Berlin Heidelberg New York, 1992.

[SB94] C. Sibertin-Blanc. Cooperative nets. In Robert Valette, editor, *15th International Conference on the Application and Theory of Petri Nets*, number 815 in Lecture Notes in Computer Science, pages 471–490, Berlin Heidelberg New York, 1994. Springer-Verlag.

[Val96] Rüdiger Valk. On Processes of Object Petri Nets. Fachbereichsbericht FBI-HH-B-185/96, University of Hamburg, Department for Computer Science, Vogt-Kölln Str. 30, 22527 Hamburg, Germany, June 1996.

[Val98] Rüdiger Valk. Petri Nets as Token Objects: An Introduction to Elementary Object Nets. In Jörg Desel, editor, *19th International Conference on Application and Theory of Petri nets*, number 1420 in LNCS, Berlin, 1998. Springer-Verlag.