How to Concretize Norms in NMAS? An Operational Normative Approach Presented with a Case Study from the Television Domain

Carolina Felicíssimo[1,2], Jean-Pierre Briot[2,1], Caroline Chopinaud[2], Carlos Lucena[1], José Viterbo[1]

[1] DI, PUC-Rio, Rua Marquês de São Vicente, 225, Gávea, RJ, 22453-900, Brasil
{cfelicissimo,lucena,viterbo}@inf.puc-rio.br
[2] LIP6, Paris VI, 104 Avenue du Président Kennedy, 75016, Paris, France
{jean-pierre.briot,caroline.chopinaud}@lip6.fr

Abstract

A major challenge in the research of multiagent systems (MAS) is the design and implementation of open MAS in which norms can be effectively applied to their entities and easily managed. Open MAS can be extremely dynamic due to heterogeneous agents that migrate among those systems in order to obtain resources or services not found locally. In this scenario, it is not reasonable to expect that *foreign* agents will know in advance all the norms of the MAS in which they will execute. Thus, we present in this paper our information module (named DynaCROM) that makes application agents aware of the norms they are bound to in a given moment. For this, the system developer should follow the defined DynaCROM methodology (also presented in the paper) to concretize norms in his MAS. Notwithstanding that a regulated system should have its norms enforced, an integration of DynaCROM with a mechanism for norm enforcement (named SCAAR) is also presented. A case study from the television domain illustrates the proposal.

1 Introduction

With the Web evolving towards a Semantic Web [2], it is believed that available information will be presented in a meaningful way for allowing not only humans to process its content, but also (*software*) agents. In this scenario, it is reasonable to expect that multiagent systems (MAS) will be open, permitting agents to migrate among them for obtaining resources and/or services not found in their original MAS.

Openness has led to software systems that have no centralized control and that are formed of autonomous entities [20]. Key characteristics of such systems are heterogeneity, conflicting individual goals and limited trust [1]. We assume that a MAS is an example of open system in which the action of heterogeneous, self-interested agents may deviate from the expected behavior in a context.

In order to prevent malicious actions and to build agent trust in a MAS, norms can be used for defining which actions are *permitted*, *obliged* and *prohibited* to be performed by agents so that the system does not reach an undesirable state. Agents should be able to take into account the existence of social norms in their decisions (either to follow or violate a norm) and to react to violations of the norms by other agents [5]. Norms can also be viewed as event-driven rules that trigger under appropriate conditions of events happening and, by doing so, create, update or cancel commitments affecting a set of agents [15].

Normative MAS (NMAS) as an area of research has become a major issue in the field of MAS and it can be situated at the intersection of normative systems and MAS. A NMAS is a system that conforms to or is based on norms [3]. Important works concerning NMAS (e.g., [9], [11], [23]) have been proposed recently. However, these solutions usually consider norms with a valid universal meaning in an application domain; do not support the direct design and implementation of norms specific to the application domain (e.g., political, economical, religious norms); do not support the management of norms at system execution (i.e., norm description off-line and norm enforcement on-line); and expect that agents already must be aware of the (*predefined*) system norms.

In order to remedy the drawbacks listed above, we propose to extend the notion of NMAS with an extra layer – called a *contextual normative level* – in which norms are embodied with domain values according to agents' current contexts. Our proposition follows the ideas first proposed by Dignum in [8] and, then, refined in [18]. However, these works mainly address formal issues while our work the practical ones, providing an implemented solution as a proof-of-concept of our ideas.

Our DynaCROM approach (meaning *Dynamic Contextual Regulation information provision in Open MAS*) [14] is, from the individual agents' perspective, an information mechanism that makes application agents aware of the norms they are bound to at a given moment. In this way, agents can simply be concerned with the applicable regulation information and, so, released from knowing in advance all the

norms of the NMAS in which they will execute. From the system developers' perspective, DynaCROM is a methodology that facilitates the tasks of design, implementation, integration and management of norms in NMAS.

In short, DynaCROM supports the implementation of *concrete norms* contextualized in NMAS. *Contextual norms* are norms described with more precise details because they are bound to the context that prescribe an application domain. We consider *context* as any implicit information that can be used to characterize the situation of agents and to provide relevant information and/or services to them, where relevancy depends on agent tasks [7].

In this paper, we illustrate the use of our DynaCROM approach for norm appliance in MAS via a motivating scenario from the television domain[1]. In the scenario, agents are self-efficient in terms that they perform their tasks without the necessity to interact with other agents from the domain. In this way, some solutions for norm enforcement (e.g., [23] and [11]) do not properly work because their regulation is restricted to the interaction level. Thus, we chose the SCAAR solution (meaning *Self-Controlled Autonomous Agents geneRator*) [6] for enforcing DynaCROM norms. SCAAR enhances agents with a self-monitoring capability that avoids norm violation in different levels of regulation.

The remainder of this paper is organized as follows. Section 2 briefly describes the main assets of DynaCROM necessary for the understanding of the paper and, in section 3, the use of these assets are exemplified via a motivating scenario from the television domain. For the domain, a DynaCROM NMAS is proposed. Section 4 explains how openness is guaranteed in a DynaCROM NMAS and section 5 shows how its norms can be enforced. Before concluding and presenting future work, Section 6 positions our work with respect to two other approaches.

2 Norm Information Provision in MAS

A major challenge in NMAS is how norms can be effectively applied to their agents and easily managed. These tasks are arduous because norms are usually written for general purposes, hindering a more precise regulation.

DynaCROM proposes to system developers a methodology for norm appliance and management in NMAS. This methodology guides them to embody abstract norms with contextualized domain values in order to refine regulation information. Following this methodology, the system developer should define the norms of his NMAS and, then, classify those in the context in which they apply in the application domain. For instance, the norm "marriage is only valid between *Jewish* people" should be classified as a *Judaism* norm contextualized in the *religious* domain.

For representing norms contextualized in application domains, DynaCROM proposes a *contextual normative ontology* in which information is provided to heterogeneous agents with a common understanding about well-defined system regulation. An *ontology* is a conceptual model that embodies shared conceptualizations of a given domain [19]; a *contextual ontology* is an ontology that represents localized domain information [4] (e.g., USD is the national currency of USA); and a *contextual normative ontology* is a contextual ontology that has a *Norm* concept as a central asset. This concept should be instantiated with norms contextualized according to basic MAS entities (i.e., environments, organizations, roles and agent interactions [22]) or specific domain entities.

The DynaCROM ontology can have its concepts composed, resulting in sets of independent norms in which the semantic of one norm can influence the semantics of the others. For the composition process, the system developer has to write *ontology-driven rules* (i.e., rules written according to the ontology structure) and, then, DynaCROM automatically infers composed contextual information.

For the composition process, DynaCROM uses an inference rule engine[2], which executes the tasks: (i) read an ontology instance to get data (i.e., concept instances and relationships), (ii) read a rule file to get how concepts must be composed, and (iii) infer an ontology instance based on the previous readings. An overview of this process is illustrated in Fig. 1.

DynaCROM is currently implemented as an active behavior[3] for continuously getting agents' updated contexts and their respective norms. All present norms are norms applicable at a given moment. Once the domain ontology and/or rule file change, updated norms are automatically forwarded to agents in the next DynaCROM execution. The dynamics of the process is an important asset of DynaCROM compared with other normative solutions (e.g., [6], [11], [23]) because it permits the management of norms at the system runtime, providing the flexibility necessary for regulation regarding social changes characteristic of MAS.

[1] See in [13] another case study with a motivating scenario from the domain of multinational corporations and focusing on agents' decisions based on contextual norms.

[2] Currently, the Jena reasoner engine is used in the DynaCROM implementation, but other Semantic Web reasoners, like Racer, Pellet or FaCT, can also be used.

[3] DynaCROM is implemented in JADE [21], but it can also be applied in other platforms by implementing their facade design pattern provided for the agent unit.

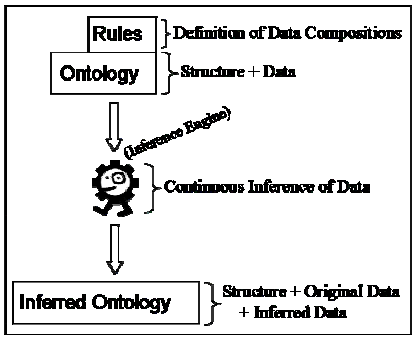

Fig. 1. The DynaCROM composition process

3 A Television Normative MAS

In order to illustrate our DynaCROM approach, we propose a scenario from the television (TV) domain. In the scenario, data can be distributed among several countries by broadcaster agents from different TV corporations. A hierarchy exists among the concepts of the example, providing implicit contextual information for regulation in NMAS.

The conceptual model illustrated in Fig. 2 was conceived in OWL in order to represent our motivating scenario. There, a *TVBroadcaster* (a type of *Organization*) is subordinated to its respective *TVCorporation* via the *isMemberOf* property, and each TV broadcaster has only one headquarters (*hasHeadquarters* inherited property). A *TVBroadcaster* is in an *Environment* (*isIn* inherited property). An *Environment* is linked to its *Government* via the *hasGovernment* property. Each *Government* is linked to its *AudioAndMediaGovernmentAgency* via the *hasGovernmentAgency* property.

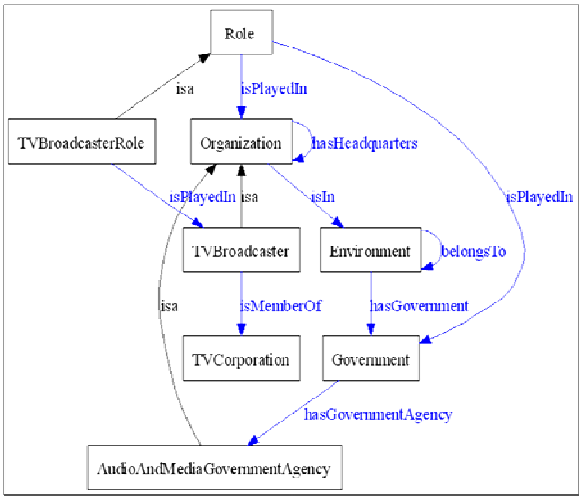

Fig. 2. A conceptual model for norm appliance in TV MAS

All concepts of the proposed conceptual model can be linked to a *Norm* concept via a *hasNorm* property for holding the norms of our proposed NMAS. In this way, norms are contextualized in each concept of the domain. For instance, we can consider, besides others, the following norm of censorship to screen a TV program:

> *A Television Norm for Censorship*: television programs are obliged to present their television rating information for giving spectators an idea of the suitability of the program for children and/or adults.

This norm is an abstract one and, thus, must be translated to a concrete norm according to the context in which it applies in the TV domain. Considering the Australian TV system, for instance, the norm can be contextualized differently in the Australian Broadcasting TV corporation (ABC) and in the Special Broadcasting Service TV corporation (SBS) – two governmental TV corporations – and also in the Nine TV corporation (Nine) – a commercial one – as follows:

> *(Concrete) Television Norms for Censorship*: (a) In the *ABC* and *SBS* TV corporations, TV ratings are obliged to be presented as follows: for ages of (i) *12* and up, (ii) *14* and up, and (iii) *18* and up; (b) in the *Nine* TV corporation, TV ratings are obliged to be presented as follows: for ages of (i) *15* and up, and (ii) *18* and up.

TV programs are screened by broadcaster agents playing the role in broadcaster organizations, meaning that the norm for censorship must be applied to all of them. The SBS TV corporation has only one broadcaster, the SBS TV broadcaster, which is situated in the state of New South Wales. Excluding the Tasmania state, which does not have a broadcaster for the Nine TV corporation, all other Australian states have broadcasters for the ABC and Nine TV corporations (some of them presented in Table 1).

Broadcaster agents can be directly influenced by the norms of its organization (via the *isPlayedIn* property) and/or indirectly influenced by the norms of its environments (e.g. city, state, country), government, government agency and TV corporation.

Table 1. TV broadcasters of Australian TV corporations

TV Broadcaster	TV Corp.	State	City
ABN TCN	ABC Nine	New South Wales	Sydney Willoughby
ABV GTV	ABC Nine	Victoria	Melbourne
ABT ---	ABC Nine	Tasmania	Hobart
ABN TCN	ABC Nine	Australian Capital Territory	Canberra
ABD NTD	ABC Nine	Northern Territory	Darwin

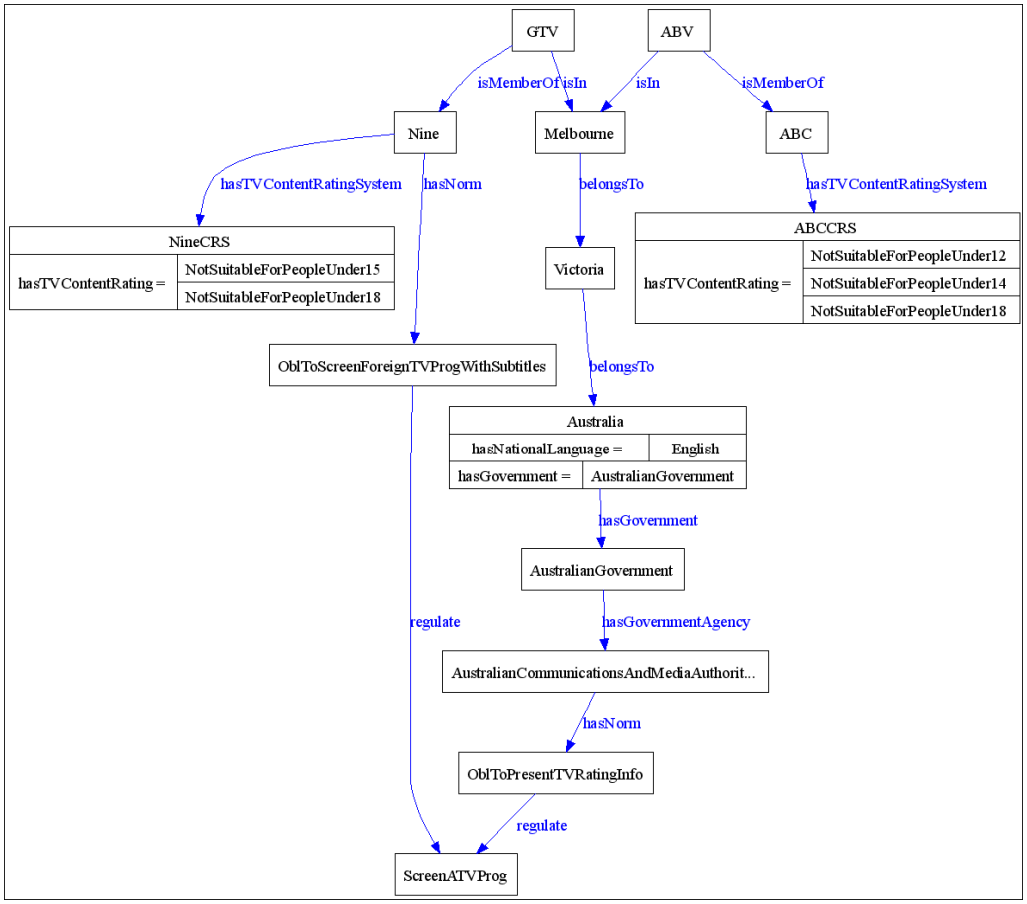

Fig. 3. An instance part of the conceptual model for norm appliance and management in the proposed television NMAS

Considering basic criteria for the development of software, it is not reasonable to expect that the censorship norm of each TV corporation will be implemented inside all of its broadcasters. So, the norm is represented by a government agency norm (illustrated in Fig. 3 by the *OblToPresentTVRatingInfo* norm instance) that links the government agency and the *ScreenATVProg* action instances.
Rules are used for applying the censorship norm in all TV broadcasters from the domain. For readability purposes, Code 1 presents those rules written following a simplified syntax for rule inference engines.

Code 1. Applying the censorship norm in broadcasters

```
(1) [DynaCROMRule_GovWithGovAgencyNorms:
(2)   hasNorm(?Gov,?GovAgencyNorms)
(3)   <- hasNorm(?GovAgency,?GovAgencyNorms),
(4)      hasGovernmentAgency(?Gov,?GovAgency)]

(5) [DynaCROMRule_EnvWithGovNorms:
(6)   hasNorm(?Env,?EnvGovNorms)
(7)   <- hasNorm(?EnvGov,?EnvGovNorms),
(8)      hasGovernment(?Env,?EnvGov)]

(9) [DynaCROMRule_EnvWithOEnvNorms:
(10)  hasNorm(?Env,?OEnvNorms)
(11)  <- hasNorm(?OEnv,?OEnvNorms),
(12)     belongsTo(?Env,?OEnv)]

(13)[DynaCROMRule_OrgWithEnvNorms:
(14)  hasNorm(?Org,?OrgEnvNorms)
(15)   <- hasNorm(?OrgEnv,?OrgEnvNorms),
(16)      isIn(?Org,?OrgEnv)]
```

The '*DynaCRORule_GovWithGovAgencyNorms*' (line 1 to 4) states that a given government will have its norms composed with the norms of its government agency. For instance, the following process is executed for the *AustralianGovernment* given value: in (4), the '*?GovAgency*' variable is instantiated with the *AustralianCommunicationAndMediaAuthority* inferred value; in (3), the '*?GovAgencyNorms*' variable is instantiated with the *OblToPresentTVRatingInfo* inferred value; and in (2), this norm is added to the *AustralianGovernment* instance.
Following a similar composition process, the '*DynaCROMRule_EnvWithGovNorms*' (line 5 to 8) states that a given environment will have its norms composed with the norms of its government; the '*DynaCROMRule_EnvWithOEnvNorms*' (line 9 to 12) states that hierarchical environments will have their norms composed; and, the '*DynaCROMRule_OrgWithEnvNorms*' (line 13 to 16) states that a given organization will have its norms composed with the norms of its environment.

The final result of the inference process is that all broadcasters of the domain are obliged to present the TV ratings of their corporations to every broadcasted program. For instance, when an agent is playing the broadcaster role in a broadcaster organization member of ABC (e.g., in ABV), the agent is automatically informed by DynaCROM about the censorship norm and the TV content ratings inherited from ABC (i.e., for ages of 12, 14 or 18, and up).

In a DynaCROM NMAS, an action can be regulated by one or more contextual norms. Another norm that can be considered in the TV domain, while screening a TV program, is as follows:

> *An (Abstract) TV Corporation Norm to Screen a Foreign Television Program*: foreign television programs are obliged to be screened with subtitles in the national language of the country in which they are broadcast. *A (Concrete) TV Corporation Norm to Screen a Foreign Television Program*: (a) broadcasters of the *Nine* TV corporation are obliged to screen foreign programs with subtitles in *English*.

In the application domain of our motivating scenario, it was chosen that only the *Nine* TV corporation holds the norm presented above. Thus, the norm is represented by the *OblToScreenForeignTVProgWithSubtitles* norm instance that links the *Nine* TV corporation and the *ScreenATVProg* action, all instances also illustrated in Fig. 3.

Code 2 presents a DynaCROM rule that apply the norm in TV broadcasters. Because the norm is only presented in the *Nine* TV corporation, thus, its broadcasters are the unique affected by the verification of line 3. For instance, considering *GTV (*a *Nine* TV broadcaster), the following process is executed: in (4), the '*?TVCorp*' variable is instantiated with the *Nine* inferred value when the '*?TVBroadcaster*' variable is instantiated with the *GTV* given value; in (3), the '*?TVCorpNorms*' variable is instantiated with the *OblToScreenForeignTVProgWithSubtitles* inferred value; and in (2), this norm is added to the *GTV* instance.

Code 2. Applying corporation norms in their broadcasters

```
(1)[DynaCROMRule_TVBroadcasterWithTVCorpNorms:
(2) hasNorm(?TVBroadcaster,?TVCorpNorms)
(3)  <- hasNorm(?TVCorp,?TVCorpNorms),
(4)     isMemberOf(?TVBroadcaster,?TVCorp)]
```

A similar inference process (distinguished only by the input values for the '*?TVBroadcaster*' variable) results in the automatic appliance of the norm in the others TV broadcasters of the *Nine* TV corporation.

4 Openness in a DynaCROM NMAS

Agents executing in NMAS are heterogeneous, implemented by different third-party developers, with code that is inaccessible. A viable solution for regulation in NMAS should not be hard coded inside agents' original codes and it must allow some flexibility for updating data (e.g., norms) during the system execution [17].

Openness is achieved in a DynaCROM NMAS because only abstract classes and methods for the domain are specified by the system developer. Agents need to choose which role they will play in the NMAS and, according to a chosen role, they need to implement the roles' respective methods (i.e., agents only need to know how to act in the NMAS in which they will perform). The methods from the NMAS can be freely implemented by agent developers with the only condition being to use the exact names given by the system developer for the defined entry parameters and returned values. Thus, the collective acceptance of developers when implementing their agents is necessary in order to make a NMAS a real application.

Code 3 exemplifies part of an abstract class and its method to be implemented by broadcaster agents in order to screen a TV program in our NMAS. Agents can request domain data to DynaCROM for correctly executing the method. For this, the predefined DynaCROM method named *getDynaCROMInfo(String infoRequested)* should be called by agents with the *infoRequested* ontology parameter filled with the desired information. DynaCROM returns the parameter values based on the ontology concepts that have their attributes matching the mask "has + infoRequested".

For instance, in the *screenATVProg()* method from Code 3, the information about the values of the *TVContentRatingSystem* (line 5) and *NationalLanguage* (line 6) parameters are provided by DynaCROM via the *getDynaCROMInfo (TVContentRatingSystem)* and *getDynaCROMInfo(NationalLanguage)* respective method calls. Considering an agent playing the broadcaster role in the *GTV* broadcaster organization, as an example, the agent is informed about a list with the concrete elements (*NotSuitableForPeopleUnder15*, *NotSuitableForPeopleUnder18*) contextualized according to the *Nine* TV content rating system, and a string with the *English* concrete value contextualized according to the *Australia* country.

Code 3. A method to be implemented by broadcaster agents

```
(1)public abstract class PlanForBroadcasterAgt{
(2) public Object screenATVProg(){
(3)  Object agtTVProg [String TVRatingInfo,
                              String SubtileIn, …];
(4)  agtTVProg = chooseATVProgram();
(5)  agtTVProg.TVRatingInfo = getTVRatingInfo(
               agtTVProg,TVContentRatingSystem);
(6)  agtTVProg.SubtileIn = getSubtitle(
                         agtTVProg,NationalLanguage);
(7)… return agtTVProgram; } }
```

DynaCROM infers domain data and applies it in its concepts following the composition process illustrated in Fig. 1. In order to allow the process, the system developer should write the rules for data composition and, then, according to these rules and the domain ontology instance, DynaCROM automatically infers data. For instance, Code 4 presents a rule for adding the information about the TV content rating system of each TV corporation from the domain ontology in its broadcasters.

Code 4. Applying content rating systems in broadcasters

```
(1)[DynaCROMRule_TVBroadCWithContentRatSystem:
(2) hasTVContentRatingSystem(?TVBroadcaster,
                                    ?TVCorpCRS)
(3)  <- hasTVContentRatingSystem(?TVCorp,
                                    ?TVCorpCRS),
(4)     isMemberOf(?TVBroadcaster,?TVCorp)]
```

The result of the inference process is that all the TV broadcasters of the *ABC* TV corporation are informed about the *ABC* TV content rating system (according to the values in the *ABCCRS* instance from the domain ontology), the TV broadcasters of the *Nine* TV corporation are informed about the *Nine* TV content rating system (according to the values in the *NineCRS* instance), and the *SBS* TV broadcaster is informed about the *SBS* TV content rating system. Once any information is updated in the content rating system of a TV corporation (e.g., the *NotSuitableForPeopleUnder13* new TV content rating information value is inserted in the *Nine* TV content rating system), it is automatically forwarded to its broadcasters (e.g., *TCN* and the others from Table 1) in the next execution of DynaCROM, without the need to restart the system. Likewise, it happens if any value is deleted or changed in the domain ontology or new compositions of customized rules are created.

Code 5 presents another example of rules for composing domain data. The '*DynaCROMRule_EnvWithOEnvNationalLangInfo*' (line 1 to 4) states that hierarchical environments will receive the information about the national language of their countries; and, the '*DynaCROMRule_TVBroadcasterWithNationalLang*' (line 5 to 8) states that a given broadcaster will also receive the same information.

Code 5. Appling national languages in broadcasters

```
(1)[DynaCROMRule_EnvWithOEnvNationalLangInfo:
(2) hasNationalLanguage(?Env,?OEnvNatLang)
(3)  <- hasNationalLanguage(?OEnv,?OEnvNatLang),
(4)     belongsTo(?Env,?OEnv)]

(5)[DynaCROMRule_TVBroadcasterWithNationalLang:
(6) hasNationalLanguage(?TVBroadcaster,
                                    ?EnvNatLang)
(7)  <- hasNationalLanguage(?Env,?EnvNatLang),
(8)     isIn(?TVBroadcaster,?Env)]
```

The same rules could be exactly used (without any change) in the case that, in the domain, besides Australian corporations, exist corporations from other countries, for instance, a Brazilian corporation. The result would be that TV broadcasters are obliged to screen foreign programs with subtitles in *Portuguese* (the national language of *Brazil*).

5 Contextual Norm Enforcement

DynaCROM is an approach for implementing dynamic NMAS in which norms can be updated at system runtime, and also for continuously supporting agents with precise information about their current norms. Nevertheless, a regulated NMAS should verify if a performed action is legal or illegal based on its defined norms, which should also be enforced. Therefore, experiments were done integrating DynaCROM with SCAAR and MOSES, two solutions for norm enforcement. In SCAAR [6] the norm enforcement is based on agents' internal behaviors; in MOSES [23] it is based on agents' external behaviors. For both solutions, DynaCROM works providing precise norm information as their input. As the enforcement solution is not the focus of DynaCROM, so, we do not deal with problems related to this part (e.g., malfunction of the enforcer).

In the following sub-section, we will describe the integration of DynaCROM and SCAAR, leaving the one with MOSES to be detailed in a future publication. SCAAR was chosen for working with DynaCROM because its mechanism is not restricted to the interaction level, permitting the enforcement of norms in other levels of abstractions (e.g., in the level of TV broadcasters).

DynaCROM and SCAAR Working Together

SCAAR is a norm enforcement mechanism that enhances agents with a self-monitoring capability for avoiding norm violation. SCAAR utilizes control hooks to trigger an enforcement core each time a regulated action occurs. When agents spontaneously incorporate the DynaCROM behavior, aiming to receive updated system norms, they also incorporate SCAAR. In the incorporation process, DynaCROM automatically replaces the headers of the regulated methods implemented inside agents, which have their signature predefined by DynaCROM, to the methods enhanced with the SCAAR control hooks.

Control hooks can be inserted inside agents' code before a regulated action, for preventing norm violation, or after, for detecting norm violation. This is a decision of the system developer when implementing the DynaCROM headers to be replaced in agents' codes. Once a regulated action start running, its control hook triggers the agent enforcement core for the verification and/or enforcement of norm compliance. Norms are represented by Petri nets [24] for verification of compliance, and inhibitor arcs are used to permit the norm enforcement.

If the system developer decides to use the SCAAR norm prevention mechanism in his NMAS, then, when a tentative of violation occurs with an obligation or prohibition norm, the enforcement core blocks the execution of the infringing action and informs it to DynaCROM; if the system developer decides to only use the SCAAR norm detection mechanism, then, when a norm violation occurs with an obligation or prohibition norm, the enforcement core informs it to DynaCROM. For a permitted norm, no specific action is taken by SCAAR.

For DynaCROM and SCAAR work together properly, the system developer should write the abstract norms of his system according to both the SCAAR syntax and the DynaCROM domain ontology's structure. Then, these norms must be concretized with instance values in the domain ontology. Concrete actions and norms must be written with the same names in both SCAAR norms and DynaCROM domain ontology. For the enforcement process, DynaCROM reads the ontology and rule files to automatically instantiate the abstract norms with domain values (see Fig.

1), providing concrete norms as input to SCAAR. SCAAR considers norms written according to the following definition, in which each term represents a set of clauses.

Norm Definition.
N (a DO A [AND P]) [IF (a BE in S [AND P])]

```
N := OBLIGED | FORBIDDEN
a := an agent playing a system role
A := an action expression
P := a proposition concerning A or S
S := a state
```

The exact identification of parameters and their attributes are predefined in the specification of the abstract classes and methods of a DynaCROM NMAS and, so, must be respected by the system developer while writing SCAAR norms. For instance, Code 6 presents an example of a SCAAR norm that the system developer wrote to regulate the broadcaster agents from his NMAS while they screen a TV program (i.e., to regulate the *ScreenATVProg* action instance from the DynaCROM domain ontology). A broadcaster agent informs its TV program via the '*agtTVProgram*' parameter (line 2). Then, SCAAR verifies in the parameter if the value of its '*TVRatingInfo*' attribute is valid, i.e. if the value proposed by the agent is one of the content rating of the TV broadcaster in which the agent is.
The list with all TV content ratings of a specific TV broadcaster is automatically provided by DynaCROM by executing progressive inferences, as exemplified in Code 7 considering the agent in the *GTV* broadcaster organization. DynaCROM instantiates the '(*domainTVBT.hasTVContentRatingSystem).hasTVContentRating*' variable applying the rule from Code 4 to all broadcasters from the domain ontology that fulfill the norm condition inherited from their related instances (following the rules from Code 1). The norm is enforced each time that it is presented in the analyzed TV broadcaster due to the verification occurred in the lines 3 and 4 of the Code 6.

Code 6. Enforcing the obligation to present tv rating information

```
(1)SCAARNorm_OblToPresentTVRatingInfo:
(2)[OBLIGED (agt DO ScreenATVProg(agtTVProg) AND
((agtTVProg.TVRatingInfo) isIn ((domainTVBT.has-
     TVContentRatingSystem).hasTVContentRating))
(3) IF (agt BE in TVBroadcaster AND
                          (agtTVBT == domainTVBT)
(4)  AND ("OblToPresentTVRatingInfo" isIn
                           (domainTVBT.hasNorm))]
```

Code 7. Part of a SCAAR norm concretized by DynaCROM

```
(2)…((agtTVProg.TVRatingInfo) isIn ((GTV).hasTV-
      ContentRatingSystem).hasTVContentRating))
   …
      (NineCRS).hasTVContentRating)
   …
      (NotSuitableForPeopleUnder15,
                NotSuitableForPeopleUnder18) …]
```

Code 8 presents another example of a SCAAR norm. The norm was written by a system developer in order to regulate the broadcaster agents from his NMAS while they screen a foreign TV program. A broadcaster agent informs its TV program via the '*agtTVProgram*' parameter (line 2). Then, SCAAR verifies, in the parameter, if the value of its '*SubtitleIn*' attribute is equal to the expected value of the '*domainTVBT.hasNationalLanguage*' variable.
DynaCROM instantiates the '*domainTVBT*' and '*domainTVBT.hasNationalLanguage*' variables applying the rules from Code 5 to all broadcasters from the domain ontology that fulfill the norm condition inherited from their related corporations (following the rule from Code 2). For instance, DynaCROM instantiates the '*domainTVBT.hasNationalLanguage*' variable with the *English* value when the agent is in the *GTV* broadcaster. The norm is inherited from *Nine*, the hierarchical TV corporation of *GTV*.

Code 8. Enforcing the obligation to present subtitles

```
(1)SCAARNorm_OblToScreenFTVProgWithSubtitles:
(2)[OBLIGED(agt DO ScreenATVProg(agtTVProg) AND
(agtTVProg.SubtitleIn == dTVBT.
                              hasNationalLanguage)
(3) IF(agt BE in TVBroadcaster AND
(agtTVBT == dTVBT) AND (agtTVProg.isForeign) AND
(4) ("OblToScreenForeignTVProgWithSubtitles"
                            isIn (dTVBT.hasNorm))]
```

DynaCROM may use an external thesaurus, as the WordNet one [27], for giving other possibilities as input to the SCAAR enforcement [12]. In this way, when the "*English language*" value is given by a regulated agent instead of simply the "*English*" value, for instance, then, SCAAR correctly infers that the agent is compliant to the norm.

6 Related Work

Electronic Agent-Based Organizations. The OMNI framework (meaning *Organizational Model for Normative Institutions*) is proposed in [26] for modeling agent organizations. Currently, OMNI does not provide a solution for the implementation and integration of its specifications in a given NMAS. Thus, DynaCROM can provide a flexible solution for implementing agent organizations by representing the OMNI scenes, roles and group of roles in its ontology. Furthermore, this ontology also can be freely enriched with domain concepts and other particular fields for any concept. The integration of organizational data in the NMAS transparently occurs when agents incorporate DynaCROM and, then, start receiving domain information.
Electronic Agent-Based Institutions. Electronic Institutions [10], or simply EI, are agent-based institutions with their focus on the institutional aspect of organizations. An EI can be specified and verified by using the ISLANDER [9] graphical tool and it uses the AMELI [11] agent-based middleware as an infrastructure that mediates agents' interactions while enforcing institutional norms.
DynaCROM can be used in AMELI by feeding governor agents with precise norm information according to agents' current contexts, or it can be used in EI in the place of AMELI for enforcing institutional norms. The main advantage in using DynaCROM as the EI enforcement mechanism is that the great number of messages exchanged between agents and their respective governors, and between

governor agents and scene manager agents is minimized. This is because, with DynaCROM, each regulated agent is enhanced with an enforcement core responsible for enforcing the system norms. Yet, agents are relieved to know in advance all the norms of the EI in which they will play.

7 Conclusion

The motivating question of our research is how norms can be effectively applied to the entities of NMAS and how they can be easily managed. Our ongoing work, named DynaCROM, intends to be a straightforward method for smoothly applying and managing regulations in NMAS as well as for enforcing precise contextual norms.

DynaCROM is still a work in progress, but we agree that it already has contributions for the domain of NMAS. DynaCROM's main contributions are: (i) a contextual normative ontology to explicitly represent the semantics of classified norms; (ii) a definition of a norm composition process that makes it easy to update regulation at system runtime; and (iii) a solution for enforcing contextual norms.

DynaCROM is not tightly coupled with a particular enforcement mechanism. In this paper, we present the first results of the integration of DynaCROM and SCAAR for enforcing the contextual norms of a NMAS from the television domain. We also started an experiment using the MOSES norm enforcement mechanism. For future work, we will continue the experiments with SCAAR and MOSES in order to compare their results. Moreover, we intend to analyze how well-founded inputs of DynaCROM can minimize conflicting norms in NMAS.

References

[1] Artikis, A., Pitt, J., Sergot, M.: Animated specifications of computational societies, In: AAMAS, 2002, pp. 1053–1061.

[2] Berners-Lee, T.; Hendler, J.; Lassila, O.: The Semantic Web, Scientific American. 284(5), p. 34–43, May 2001.

[3] Boella, G.; Torre, L.; Verhagen, H.: Introduction to normative multiagent systems, In: Journal of Comput. Math. Organ. Theory, 12(2-3), 2006, pp. 71–79.

[4] Bouquet, P., Giunchiglia, F., Harmelen, F.v., Serafini, L., Stuckenschmidt, H.: C-OWL: Contextualizing Ontologies, In: ISWC-03, LNCS 2870, 2003, pp. 164–179.

[5] Castelfranchi, C., Dignum, F., Jonker, C.M., Treur, J.: Deliberative Normative Agents: Principles and Architecture, 1999, in: ATAL-99, 1999, pp. 364–378.

[6] Chopinaud, C., Seghrouchni, A.E.F., Taillibert, P.: Prevention of harmful behaviors within cognitive and autonomous agents, In: ECAI'06, pp. 205–209.

[7] Dey, A.: Understanding and using context, In: Personal and Ubiquitous Computing, 5(1), 2001, 4–7, pp. 1617–4909.

[8] Dignum, F.: Abstract norms and electronic institutions, in: RASTA'02, Bologna, 2002, pp. 93–104.

[9] Esteva, M.; Cruz, D. de la; Sierra, C.: ISLANDER: an electronic institutions editor, in: AAMAS, 2002, pp. 1045–1052.

[10] Esteva, M.: Electronic Institutions: from specification to development, IIIAPhD thesis(19).

[11] Esteva, M.; Rosell, B.; Rodriguez-Aguilar, J.A.; Arcos, J.Ll.: AMELI: An Agent-based Middleware for Electronic Institutions, In: AAMAS, 2004, pp. 236 – 243.

[12] Felicíssimo, C.: Semantic Web Interoperability: One strategy for the Taxonomic Ontology Alignment. Master Dissertation from PUC-Rio. August, 2004. Brazil. http://www2.dbd.puc-rio.br/pergamum/tesesabertas/0220943_04_pretextual.pdf.

[13] Felicíssimo, C.; Choren, R.; Briot, J-P.; Lucena, C.: Informing Regulatory Dynamics in Open MAS. In: COIN 06 post proceedings, LNAI 4386, pp. 147–162, 2007.

[14] Felicíssimo, C.; Chopinaud, C.; Briot, J-P.; Seghrouchni, A.E.F.; Lucena, C.: Contextualizing Normative Open Multi-Agent Systems. In: 23º Annual ACM SAC, 2008. pp. 52-59.

[15] Fornara, N., Viganò, F., Colombetti, M.: Agent communication and artificial institutions. In: JAAMAS, 2007, (14) pp. 121–142.

[16] Gosling, J., Joy, B., Junior, G.L.S., Bracha, G.: The Java Language Specification, 2008, http://java.sun.com.

[17] Grizard, A., Vercouter, L., Stratulat, T., Muller, G.: A peer-to-peer normative system to achieve social order, in: COIN@AAMAS-2006, 2006, Japan.

[18] Grossi, D.; Dignum, F.: From Abstract to Concrete Norms in Agent Institutions, in: Proc. of the FAABS III, 2004. LNCS 3228, 2005, pp. 12–29, ISBN: 978-3-540-24422-6.

[19] Gruber, T. R.: A translation approach to portable ontology specifications. In: Knowledge Acquisition, 5 (2), 1993, pp. 199–220, pp. 1042–8143.

[20] Hewitt, C.: Open Information Systems Semantics for Distributed Artificial Intelligence, In: Artificial Intelligence, 47(1-3), 1991, pp. 79–106, ISSN: 0004-3702.

[21] JADE, 2008, http://jade.tilab.com.

[22] Jennings, N. R.: On Agent-Based Software Engineering, in: AI, 117 (2) pp. 277–296.

[23] Minsky, N.: MOSES, 2008, http://www.moses.rutgers.edu/.

[24] Murata,T.: Petri nets: Properties, analysis and applications. In: IEEE 77(4), 1989, 541–580.

[25] Searle, R.: The construction of social reality. NY Free Press.

[26] Vázquez-Salceda, J.; Dignum, V.; Dignum, F.: Organizing Multiagent Systems. JAAMAS, 2005, 11(3), pp. 307–360.

[27] The WordNet Thesuarus online, 2008, http://wordnet.princeton.edu/perl/webwn.