A Logical Architecture of a Normative System

Guido Boella[1] and Leendert van der Torre[2]

[1] Dipartimento di Informatica Università di Torino, Italy E-mail: guido@di.unito.it.
[2] University of Luxembourg. E-mail: leendert@vandertorre.com

Abstract. Logical architectures combine several logics into a more complex logical system. In this paper we study a logical architecture using input/output operations corresponding to the functionality of logical components. We illustrate how the architectural approach can be used to develop a logic of a normative system based on logics of counts-as conditionals, institutional constraints, obligations and permissions. In this example we adapt for counts-as conditionals and institutional constraints a proposal of Jones and Sergot, and for obligations and permissions we adapt the input/output logic framework of Makinson and van der Torre. We use our architecture to study logical relations among counts-as conditionals, institutional constraints, obligations and permissions. We show that in our logical architecture the combined system of counts-as conditionals and institutional constraints reduces to the logic of institutional constraints, which again reduces to an expression in the underlying base logic. Counts-as conditionals and institutional constraints are defined as a pre-processing step for the regulative norms. Permissions are defined as exceptions to obligations and their interaction is characterized.

1 Introduction

In this paper we are interested in logical architectures. The notion of an 'architecture' is used not only in the world of bricks and stones, but it is used metaphorically in many other areas too. For example, in computer science it is used to describe the hard- or software organization of systems, in management science it is used to describe the structure of business models in enterprise architectures [16], and in psychology and artificial intelligence it is used to describe cognitive architectures of agent systems like SOAR [15], ACT [2] or PRS [13]. Though architectures are typically visualized as a diagram and informal, there are also various formal languages to describe architectures, see, for example, [16]. The notion of architecture reflects in all these examples an abstract description of a system in terms of its components and the relations among these components. This is also how we use the metaphor in this paper. In logic and knowledge representation, architectures combine several logics into a more complex logical system.

Advantages of the architectural approach in logic are that logical subsystems can be analyzed in relation to their environment, and that a divide and conquer strategy can reduce a complex theorem prover to simpler proof systems. These advantages are related to the advantages of architectural approaches in other areas. For example, in computer science a devide and conquer strategy is used frequently to develop computer systems.

In this area, the architectural approach is used also to bridge the gap between formal specification and architectural design, and to facilitate the communication among stakeholders discussing a system, using visual architectural description languages [16].

The logical architecture we introduce and discuss in this paper is based on an architecture we recently introduced for normative multiagent systems [7]. In Figure 1, components are visualized by boxes, and the communication channels relating the components are visualized by arrows. There are components for counts-as conditionals (CA), institutional constraints (IC), obligations (O) and permissions (P). Moreover, the norm base (NB) component contains sets of norms or rules, which are used in the other components to generate the component's output from its input. This component does not have any inputs, though input channels can be added to the architecture to represent ways to modify the norms. The institutional constraints act as a wrapper around the counts-as component to enable the connection with the other components, as explained in detail in this paper. The open circles are ports or interface nodes of the components, and the black circles are a special kind of merge nodes, as explained later too. Note that the architecture is only an abstract description of the normative system, focussing on the relations among various kinds of norms, but for example abstracting away from sanctions, control systems, or the roles of agents being played in the system.

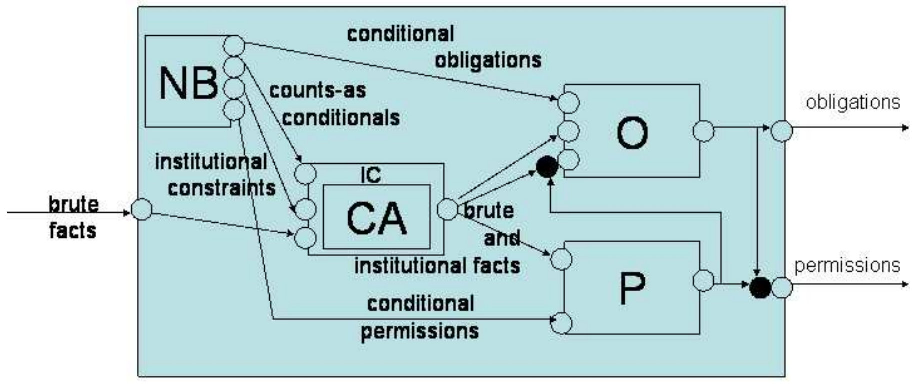

Fig. 1. A Logical Architecture of a Normative System.

Figure 1 is a visualization of a logical architecture, where logical input/output operations correspond to the functionality of components. Considering a normative system as an input/output operation is not unusual. For example, inspired by Tarski's definition of deductive systems, Alchourrón and Bulygin [1] introduce normative systems with as inputs factual descriptions and as output obligatory and permitted situations. For counts-as conditionals and institutional constraints we adapt a proposal of Jones and Sergot [14], and for obligations and permissions we adapt the input/output logic framework of Makinson and van der Torre [17]. Moreover, we use Searle's distinction between regulative and constitutive norms [21], and brute and institutional facts.

The contribution of the paper thus lies not in the components, but in their mutual integration. Rather than following some highly general template for such an integration we stick closely to the contours of the specific domain: complex normative structures with multiple components. We focus on two kinds of interactions.

The main issue in the logical architecture is the relation among regulative and constitutive norms, that is, among on the one hand obligations and permissions, and on the other hand so-called counts-as conditionals. The latter are rules that create the possibility of or define an activity. For example, according to Searle, "the activity of playing chess is constituted by action in accordance with these rules. Chess has no existence apart from these rules. The institutions of marriage, money, and promising are like the institutions of baseball and chess in that they are systems of such constitutive rules or conventions." They have been identified as the key mechanism to normative reasoning in dynamic and uncertain environments, for example to realize agent communication, electronic contracting, dynamics of organizations, see, e.g., [6].

Extending the logical analysis of the relation between constitutive and regulative norms, we also reconsider the relation between obligations and permissions in our architecture. In the deontic logic literature, the interaction between obligations and permissions has been studied in some depth. Von Wright [22] started modern deontic logic literature by observing a similarity between the relation between on the one hand necessity and possibility, and on the other hand obligation and permission. He defined permissions as the absence of a prohibition, which was later called a weak permission. Bulygin [10] argues that a strong kind of permissions must be used in context of multiple authorities and updating normative systems: if a higher authority permits you to do something, a lower authority can no longer make it prohibited. Moreover, Makinson and van der Torre distinguish backward and forward positive permissions [19]. In this paper we consider permissions as exceptions of obligations.

This paper builds on the philosophy and technical results of the input/output logic framework. Though we repeat the basic definitions we need for our study, some knowledge of input/output logic [17] or at least of its introduction [20] is probably needed. The development of input/output logic has been motivated by conditional norms, which do not have a truth value. For that reason, the semantics of input/output logic given by Makinson and van der Torre is an operational semantics, which characterizes the output as a function of the input and the set of norms. However, classical semantics for conditional norms exists too. Makinson and van der Torre illustrate how to recapture input/output logic in modal logic, and thus give it a classical possible worlds semantics. More elegantly, as illustrated by Bochman [4], the operational semantics of input/output logic can be rephrased as a bimodel semantics, in which a model of a set of conditionals is a pair of partial models from the base logic (in this paper, propositional logic).

The layout of this paper is as follows. We first represent a fragment of Jones and Sergot's logic of counts-as conditionals as an input/output operation, then we represent their logic of institutional constraints as an input/output operation, and characterize their interaction. Thereafter we adapt Makinson and van der Torre's logic of input/output for multiple constraints, and we characterize the interaction among institutional constraints and obligations. Finally we introduce permissions as an input/output operation with multiple outputs, and we use them as exceptions to obligations.

2 Counts-as conditionals (constitutive norms)

For Jones and Sergot [14], the counts-as relation expresses the fact that a state of affairs or an action of an agent "is a sufficient condition to guarantee that the institution creates some (usually normative) state of affairs". Jones and Sergot formalize this introducing a conditional connective $\Rightarrow_s$ to express the "counts-as" connection holding in the context of an institution s. They characterize the logic for $\Rightarrow_s$ as a conditional logic together with axioms for right conjunction,

$$((A \Rightarrow_s B) \wedge (A \Rightarrow_s C)) \supset (A \Rightarrow_s (B \wedge C))$$

left disjunction,

$$((A \Rightarrow_s C) \wedge (B \Rightarrow_s C)) \supset ((A \vee B) \Rightarrow_s C)$$

and transitivity.

$$((A \Rightarrow_s B) \wedge (B \Rightarrow_s C)) \supset (A \Rightarrow_s C)$$

Moreover, they consider other important properties not relevant here.

In this paper, we consider an input/output operation corresponding to the functionality of the counts-as component. This means that we restrict ourselves to the flat conditional fragment of Jones and Sergot's logic. Despite this restriction, we can still express the relevant properties Jones and Sergot discuss.[1] The input of the operation is a set of counts-as conditionals CA, a normative system or institution s, and the sufficient condition x, and an output is a created state of affairs y. Calling the operation out_{CA}, we thus write $y \in out_{CA}(CA, s, x)$. To relate the operation to the conditional logic, we equivalently write $(x, y) \in out_{CA}(CA, s)$, where (x, y) is read as "x counts as y".

Jones and Sergot propose a combined logic that incorporates besides the conditional logic also an action logic. We therefore assume a base action logic on which the input/output operation operates. This may be a logic of successful action as adopted by Jones and Sergot, according to which an agent brings it about that, or sees to it that, such-and-such is the case. But alternatively, using a simple propositional logic based on the distinction between controllable and uncontrollable propositions [9] most relevant properties can be expressed as well, and of course a more full-fledged logic of action can also be used, incorporating for example a model of causality.

Definition 1. *Let L be a propositional action logic with $\vdash$ the related notion of derivability and Cn the related consequence operation $Cn(x) = \{y \mid x \vdash y\}$. Let CA be a set of pairs of L, $\{(x_1, y_1), \ldots, (x_n, y_n)\}$, read as '$x_1$ counts as y_1', etc. Moreover, consider the following proof rules: conjunction for the output (AND), disjunction of the input (OR), and transitivity (T) defined as follows:*

$$\frac{(x, y_1), (x, y_2)}{(x, y_1 \wedge y_2)} AND \qquad \frac{(x_1, y), (x_2, y)}{(x_1 \vee x_2, y)} OR \qquad \frac{(x, y_1), (y_1, y_2)}{(x, y_2)} T$$

[1] When comparing their framework to a proposal of Goldman, Jones and Sergot mention (but do not study) irreflexive and asymmetric counts-as relations. With an extension of our logical language covering negated conditionals, these properties can be expressed too.

For an institution s, the counts-as output operator out_{CA} is defined as closure operator on the set CA using the rules above, together with a silent rule that allows replacement of logical equivalents in input and output. We write

$$(x, y) \in \mathit{out}_{\text{CA}}(CA, s)$$

Moreover, for $X \subseteq L$, we write

$$y \in \mathit{out}_{\text{CA}}(CA, s, X)$$

if there is a finite $X' \subseteq X$ such that $(\wedge X', y) \in \mathit{out}_{\text{CA}}(CA, s)$, indicating that the output y is derived by the output operator for the input X, given the counts-as conditionals CA of institution s. We also write $\mathit{out}_{\text{CA}}(CA, s, x)$ for $\mathit{out}_{\text{CA}}(CA, s, \{x\})$.

Example 1. If for some institution s we have $CA = \{(a, x), (x, y)\}$, then we have $\mathit{out}_{CA}(CA, s, a) = \{x, y\}$.

Jones and Sergot argue that the strengthening of the input (SI) and weakening of the output (WO) rules presented in Definition 2 are invalid, see their paper for a discussion. The adoption of the transitivity rule T for their logic is criticized by Artosi *et al.* [3]. Jones and Sergot say that "we have been unable to produce any counter-instances [of transitivity], and we are inclined to accept it". Neither of these authors consider to replace transitivity by cumulative transitivity (CT), see Definition 4.

Jones and Sergot give a semantics for their classical conditional logic based on minimal conditional models. For our reconstruction, the following questions can be asked (where the second could provide a kind of operational semantics analogous to the semantics of input/output logics [17], and the third to the semantics of input/output logic given in [4]):

- Given CA of s, and (x, y), do we have $(x, y) \in \mathit{out}_{CA}(CA, s)$?
- Given CA of s and x, what is $\mathit{out}_{CA}(CA, s, x)$?
- Given CA of s, what is $\mathit{out}_{CA}(CA, s)$?

The following theorem illustrates a simpler case.

Theorem 1. *With only the AND and T rule, assuming replacements by logical equivalents, $\mathit{out}_{\text{CA}}(CA, s, X) = \{\wedge Y \mid Y \subseteq \cup_{i=0}^{\infty} \mathit{out}^i_{\text{CA}}(CA, s, X)\}$ is calculated as follows.*
$\mathit{out}^0_{\text{CA}}(CA, s, X) = \emptyset$
$\mathit{out}^{i+1}_{\text{CA}}(CA, s, X) = \mathit{out}^i_{\text{CA}}(CA, s, X) \cup \{y \mid (\wedge X', y) \in CA, X' \subseteq \mathit{out}^i_{\text{CA}}(CA, s, X)\}$

With the OR rule the situation is more complicated due to reasoning by cases. However, as we show by Theorem 3 in Section 3, we do not need the semantics of this component to define a semantics of the whole normative system.

3 Institutional constraints

Jones and Sergot's analysis of counts-as conditionals is integrated with their notion of so-called institutional constraints. Note that the term "constraints" is used here in another way than it is used in input/output logic for handling contrary-to-duty obligations and also permissions, as we discuss in the following section, because the input/output constraints, but not the institutional constraints, impose consistency requirements. We have chosen not to avoid the possible confusion, given the existence of the institutional constraints in the literature and the appropriateness of input/output constraints for what it is actually doing.

Jones and Sergot introduce the normal KD modality D_s such that $D_s A$ means that A is "recognized by the institution s". D_s is represented by a so-called wrapper around the counts-as component in our normative system architecture. In computer science, a wrapper is a mechanism to pre-process the input and to post-process the output of a component. there are various ways to formalize this idea in a logical architecture. In this section we formalize it by stating that the input/output relation of the counts-as component is a subset of the input/output relation of the institutional constraints component.

Jones and Sergot distinguish relations of logical consequence, causal consequence, and deontic consequence. We do not consider the latter, as they are the obligations and permissions which we represent by separate logical subsystems. An institutional constraint "if x then y" is represented by $D_s(x \to y)$.

Definition 2. *Let IC be a set of pairs of L, $\{(x_1, y_1), \dots, (x_n, y_n)\}$, read as 'if x_1 then y_1', etc. Moreover, consider the following proof rules strengthening of the input (SI), weakening of the output (WO) and Identity (Id) defined as follows:*

$$\frac{(x_1, y)}{(x_1 \wedge x_2, y)} SI \qquad \frac{(x, y_1 \wedge y_2)}{(x, y_1)} WO \qquad \frac{}{(x, x)} Id$$

For an institution s, the institutional constraint output operator out$_{\text{IC}}$ is defined as closure operator on the set IC using the three rules of Definition 1, together with the three rules above and a silent rule that allows replacement of logical equivalents in input and output.

The output of the institutional constraints can be obtained by a reduction to the base logic. Whereas the logic of counts-as conditionals is relatively complicated, the logic of institutional constraints is straightforward.

Theorem 2. *out*$_{\text{IC}}(IC, s, x) = Cn(\{x\} \cup \{x \supset y \mid (x, y) \in IC\})$

Proof. *(sketch). T follows from the other rules, and the property for the remaining rules follows from results on throughput in [17].*

Counts-as conditionals and institutional constraints are related by Jones and Sergot by the axiom schema:

$$(A \Rightarrow_s B) \supset D_s(A \supset B)$$

Using our input/output operations, the combined system of counts-as conditionals and institutional constraints are thus characterized by the following bridge rule.

Definition 3. *Let $out_{\text{IC+CA}}$ be an operation on two sets of pairs of L, IC and CA, defined as out_{IC} (with all six rules discussed thus far) on the first parameter, together with the following rule:*

$$\frac{(x, y) \in out_{\text{CA}}(CA, s)}{(x, y) \in out_{\text{IC+CA}}(IC, CA, s)}$$

The following theorem shows that we can calculate the output of the joint system without taking the logic of counts-as conditionals into account.

Theorem 3. $out_{\text{IC+CA}}(IC, CA, s, X) = out_{\text{IC}}(IC \cup CA, s, X)$.

Proof. *(sketch). Clearly we have $out_{\text{IC+CA}}(IC, CA, s, X) = out_{\text{IC}}(IC \cup out_{\text{CA}}(CA), s, X)$. Since the proof rules of out_{IC} contain the proof rules of out_{CA}, the result follows.*

The limitation of Jones and Sergot approach, according to Gelati *et al.* [11], is that the consequences of counts-as connections follow non-defeasibly (via the closure of the logic for modality D_s under logical implication), whereas defeasibility seems a key feature of such connections. Their example is that in an auction if a person raises one hand, this may count as making a bid. However, this does not hold if he raises his hand and scratches his own head. There are many ways in which the logic of institutional constraints can be weakened, which we do not further consider here.

4 Obligations

There are many deontic logics, and there are few principles of deontic logic which have not been criticized. In this paper we do not adopt one particular deontic logic, but Makinson and van der Torre's framework [17] in which various kinds of deontic logics can be defined. Their approach is based on the concept of logic as a 'secretarial assistant', in the sense that the role of logic is not to formalize reasoning processes themselves, but to pre- and post-process such reasoning processes. Though a discussion of this philosophical point is beyond the scope of this paper, the idea of pre- and post-processing is well suited for the architectural approach.

A set of conditional obligations or rules is a set of ordered pairs $a \to x$, where a and x are sentences of a propositional language. For each such pair, the body a is thought of as an input, representing some condition or situation, and the head x is thought of as an output, representing what the rule tells us to be obligatory in that situation. Makinson and van der Torre write (a, x) to distinguish input/output rules from conditionals defined in other logics, to emphasize the property that input/output logic does not necessarily obey the identity rule. In this paper we also follow this convention. We extend the syntax of input/output logic with a parameter s for the institution to match Jones and Sergot's definitions.

Definition 4 (Input/output logic). *For an institution s, let O be a set of pairs of L, $\{(a_1, x_1), \ldots, (a_n, x_n)\}$, read as 'if a_1 then obligatory x_1', etc. Moreover, consider the following proof rules strengthening of the input (SI), conjunction for the output (AND),* *weakening of the output (WO), disjunction of the input (OR), and cumulative transitivity (CT) defined as follows:*

$$\frac{(a,x)}{(a\wedge b,x)}SI \quad \frac{(a,x\wedge y)}{(a,x)}WO \quad \frac{(a,x),(a,y)}{(a,x\wedge y)}AND$$

$$\frac{(a,x),(b,x)}{(a\vee b,x)}OR \quad \frac{(a,x),(a\wedge x,y)}{(a,y)}CT$$

The following output operators are defined as closure operators on the set O using the rules above, together with a silent rule that allows replacement of logical equivalents in input and output. Moreover, we write $(a,x)\in out_{\mathrm{O}}(O,s)$ to refer to any of these operations. We also write $x\in out_{\mathrm{O}}(O,s,A)$ if there is a finite $A'\subseteq A$ with $(\wedge A',x)\in out_{\mathrm{O}}(O,s)$.
out^1_{O}: SI+AND+WO (simple-minded output)
out^2_{O}: SI+AND+WO+OR (basic output)
out^3_{O}: SI+AND+WO+CT (reusable output)
out^4_{O}: SI+AND+WO+OR+CT (basic reusable output)

Example 2. Given $O=\{(a,x),(x,z)\}$ the output of O contains $(x\wedge a,z)$ using the rule SI. Using also the CT rule, the output contains (a,z). (a,a) follows only if there is an identity rule in addition (when Makinson and van der Torre call it throughput [17]).

The institutional constraints (and thus the counts-as conditionals) can be combined with the obligations using iteration.

Definition 5. *For an institution s, let $out_{\mathrm{IC+CA+O}}$ be an operation on three sets of pairs of L, IC, CA, and O, defined in terms of $out_{\mathrm{IC+CA}}$ and out_{O} using the following rule:*

$$\frac{(x,y)\in out_{\mathrm{IC+CA}}(IC,CA,s),(y,z)\in out_{\mathrm{O}}(O,s)}{(x,z)\in out_{\mathrm{IC+CA+O}}(IC,CA,O,s)}$$

Theorem 4.

$$out_{\mathrm{IC+CA+O}}(IC,CA,O,s,x)=out_{\mathrm{O}}(O,s,out_{\mathrm{IC+CA}}(IC,CA,x))$$

In case of contrary-to-duty obligations, the input represents something which is inalterably true, and an agent has to ask himself which rules (output) this input gives rise to: even if the input should have not come true, an agent has to “make the best out of the sad circumstances” [12]. In input/output logics under constraints, a set of mental attitudes and an input does not have a set of propositions as output, but a set of sets of propositions. We can infer a set of propositions by for example taking the join (credulous) or meet (sceptical), or something more complicated. In this paper we use the meet to calculate the output of the obligation component. Moreover, we extend the definition to a set of constraints. Although we need only one constraint set for Definition 7, we will need arbitrary sets in the following section in order to integrate permissions.

Definition 6 (Constraints). *For an institution s, let O be a set of conditional obligations, and let $\{C_1, \ldots, C_n\}$ be a set of sets of arbitrary formulas, which we will call the "constraint set". For any input set A, we define maxfamily(O,s,A,$\{C_1, \ldots, C_n\}$) to be the family of all maximal $O' \subseteq O$ such that out$_{\mathrm{O}}(O', A)$ is consistent with C_i for $1 \leq i \leq n$. Moreover, we define outfamily($O, s, A, \{C_1, \ldots, C_n\}$) to be the family of outputs under input A, generated by elements of maxfamily($O, s, A, \{C_1, \ldots, C_n\}$). The meet output under constraints is*

$$\textit{out}_{\mathrm{O}}^{\cap}(O, s, A, \{C_1, \ldots, C_n\}) = \cap \textit{outfamily}(O, s, A, \{C_1, \ldots, C_n\})$$

We can adopt an output constraint (the output has to be consistent) or an input/output constraint (the output has to be consistent with the input). The following definition uses the input/output constraint, because the output of the obligation component is consistent with the output of the institutional constraints.

Definition 7.

$$\textit{out}_{\mathrm{IC+CA+O}}^{\cap}(IC, CA, O, s, x) = \textit{out}_{\mathrm{O}}^{\cap}(O, s, \{\textit{out}_{\mathrm{IC+CA}}(IC, CA, s, x)\}, \textit{out}_{\mathrm{IC+CA}}(IC, CA, s, x))$$

See [17, 18] for the semantics of input/output logics, further details on its proof theory, its possible translation to modal or propositional logic, the extension with the identity rule, alternative constraints, and examples.

5 Permissions

Permissions are often defined in terms of obligations, called 'weak' permissions, in which case there are no conditional permissions in NB. When they are not defined as weak permissions, as in this paper, then the norm databse also contains a set of permissive norms [19]. Such 'strong' permissions are typically defined analogously to obligation, but without the AND rule. The reason AND is not accepted is that p as well as $\neg p$ can be permitted, but it does not make sense to permit a contradiction. Permissions are simpler than obligations, as the issue of contrary-to-duty reasoning is not relevant, and therefore we do not have to define constraints. Here we consider only the rules SI and WO. As the output of the permission component we do not take the union of the set indicated, as this would lose information that we need in the next integrative step.

Definition 8 (Conditional permission). *For an institution s, let conditional permissions P be a set of pairs of L, $\{(a_1, x_1), \ldots, (a_n, x_n)\}$, read as 'if a_1 then permitted x_1', etc. The output of the permission component is*

$$\textit{out}_{\mathrm{P}}(P, s, A) = \{Cn(x) \mid (\wedge A', x) \in P, A' \subseteq A\}$$

In the normative system, we merge the output of permission component X with the output of the obligation component Y to get the property that if something is obliged, then it is also permitted.

Definition 9. *Let $X \subseteq 2^L$ and $Y \subseteq L$. The merger of the two sets is defined as follows*

$$merge(X, Y) = \{Cn(x \cup Y) \mid x \in X\}$$

The combination of the counts-as conditionals and the permissions is analogous to the combination of the counts-as conditionals and obligations.

Definition 10.

$$\frac{(x,y) \in \mathit{out}_{\text{IC+CA}}(IC,CA,s), (y,Z) \in \mathit{out}_{\text{P}}(P,s)}{(x,Z) \in \mathit{out}_{\text{IC+CA+P}}(IC,CA,P,s)}$$

Finally, we consider the output of the normative system. For obligations, the output of the institutional constraints is merged with the output of the permissions component as defined in Definition 9. It is an extension of the output of the obligation component given in Definition 7, where we did not consider permissions yet.

Definition 11. $\mathit{out}^{\cap}_{\text{IC+CA+P+O}}(IC,CA,O,P,s,X) =$

$\mathit{out}^{\cap}_{\text{O}}(O, merge(\mathit{out}_{\text{IC+CA+P}}(IC,CA,P,s,X), \mathit{out}_{\text{IC+CA}}(IC,CA,s,X)), s, \mathit{out}_{\text{IC+CA}}(IC,CA,s,X))$

We have now defined the output of each component in the normative system architecture visualized in Figure 1, and the final step is to define the output of the whole normative system. The first output is the set of obligations as given in Definition 11. The second output is the set of permissions, which combines the output of the permission component with the output of the obligation component. The permissions of the normative system are defined as follows.

Definition 12. $\mathit{out}_{\text{IC+CA+P+O+P}}(IC,CA,P,O,s,X) =$

$$merge(\mathit{out}_{\text{IC+CA+P}}(IC,CA,P,s,X), \mathit{out}^{\cap}_{\text{IC+CA+P+O}}(IC,CA,O,P,s,X))$$

6 Further extensions

In a single component, feedback is represented by the cumulative transitivity rule, in the following sense [5]. If x is in the output of a, and y is in the output of $a \wedge x$, then we may reason as follows. Suppose we have as input a, and therefore as output x. Now suppose that there is a feedback loop, such that we have as input $a \wedge x$, then we can conclude as output y. Thus in this example, feedback of x corresponds with the inference that y is in the output of a.

Moreover, there may be feedback among components, leading to cycles in the network. As a generalization of the cumulative transitivity rule for the obligation component, we may add a feedback loop from the obligation component to the counts-as component, such that new institutional facts can be derived for the context in which the obligations are fulfilled. Likewise, we may add a feedback loop from obligation to permission. Since there is already a channel from permission to obligation, this will result in another cycle.

Another interesting feedback loop is from counts-as conditional to a new input of the norm database. In this way, the normative system can define how the normative system can be updated, see [6] for a logical model how constitutive norms can define the role and powers of agents in the normative system. This includes the creation of contracts, which may be seen as legal institutions, that is, as normative systems within the normative system [8].

A more technical issue is what happens when we create a feedback loop by connecting the permissions in the output of the normative system component to the constraints input of the obligation component.

7 Concluding remarks

In the paper we have presented a logical architecture of normative systems, combining the logics of counts-as conditionals and institutional constraints of Jones and Sergot with the input/output logics of Makinson and van der Torre. The contribution of the paper thus lies not in the components, but in their mutual integration. The results established are all fairly straightforward given familiarity with the background on input/output logics. Finally, we have discussed various ways in which the normative system can be further extended using timed streams, feedback loops and hierarchical normative systems.

The architecture presented is just one (rather natural) example, and there may be additional components, and other kinds of interactions which would also be worth studying using the same techniques. We believe that it may be worthwhile to study other logical components and other ways to connect the components, leading to other relations among counts-as conditionals, institutional constraints, obligations and permissions. An important contribution of our work is that it illustrates how such studies can be undertaken.

Besides the further logical analysis of the architecture of normative system, there are two other important issues of further research. First there is a need for a general theory of logical architectures, besides the existing work on combining logics and formal software engineering, along the line of logical input/output nets as envisioned in [20, 5]. Second, in agent based software engineering there is a need for a study whether the logical architecture developed here can be used for the design of architectures in normative multi-agent systems.

References

1. C. Alchourrón and Bulygin. *Normative systems.* Springer, 1972.
2. J.R. Anderson. *Rules of the mind.* Lawrence Ellbaum Associates, 1993.
3. A. Artosi, A. Rotolo, and S. Vida. On the logical nature of count-as conditionals. In *Procs. of LEA 2004 Workshop*, 2004.
4. A. Bochman. *Explanatory Nonmonotonic reasoning.* World Scientific, 2005.
5. G. Boella, J. Hulstijn, and L. van der Torre. Interaction in normative multi-agent systems. *Electronic Notes in Theoretical Computer Science*, 141(5):135–162, 2005.
6. G. Boella and L. van der Torre. Regulative and constitutive norms in normative multiagent systems. In *Proceedings of the Ninth International Conference on the Principles of Knowledge Representation and Reasoning (KR'04)*, pages 255–265, 2004.
7. G. Boella and L. van der Torre. An architecture of a normative system (short paper). In *Proceedings of AAMAS06*, 2006.

8. G. Boella and L. van der Torre. A game theoretic approach to contracts in multiagent systems. *IEEE Trans. SMC, Part C*, 36(1):68– 79, 2006.
9. C. Boutilier. Toward a logic for qualitative decision theory. In *Proceedings of KR'94*, pages 75–86, 1994.
10. E. Bulygin. Permissive norms and normative systems. In *Automated Analysis of Legal Texts*, pages 211–218. Publishing Company, Amsterdam, 1986.
11. J. Gelati, G. Governatori, N. Rotolo, and G. Sartor. Declarative power, representation, and mandate. A formal analysis. In *Procs. of JURIX 02*. IOS press, 2002.
12. B. Hansson. An analysis of some deontic logics. *Nôus*, 3:373–398, 1969.
13. F.F. Ingrand, M.P Georgeff, and A.S. Rao. An architecture for real-time reasoning and system control. *IEEE Expert*, 7(6), 1992.
14. A. Jones and M. Sergot. A formal characterisation of institutionalised power. *Journal of IGPL*, 3:427–443, 1996.
15. J.E. Laird, A. Newell, and Paul S. Rosenbloom. SOAR: an architecture for general intelligence. *Artificial Intelligence*, 33:1–64, 1987.
16. M. Lankhorst et al. *Enterprise Architecture At Work*. Springer, 2005.
17. D. Makinson and L. van der Torre. Input-output logics. *Journal of Philosophical Logic*, 29:383–408, 2000.
18. D. Makinson and L. van der Torre. Constraints for input-output logics. *Journal of Philosophical Logic*, 30:155–185, 2001.
19. D. Makinson and L. van der Torre. Permissions from an input/output perspective. *Journal of Philosophical Logic*, 32 (4):391–416, 2003.
20. D. Makinson and L. van der Torre. What is input/output logic? In *Foundations of the Formal Sciences II: Applications of Mathematical Logic in Philosophy and Linguistics*, volume 17 of *Trends in Logic*. Kluwer, 2003.
21. J. R. Searle. *Speech Acts: An Essay in the Philosophy of Language*. Cambridge University Press, Cambridge, United Kingdom, 1969.
22. G.H. von Wright. Deontic logic. *Mind*, 60:1–15, 1951.