Techne

Ivan J. Jureta
FNRS & Information Management
University of Namur
ijureta@fundp.ac.be

Alex Borgida
Department of Computer Science
Rutgers University
borgida@cs.rutgers.edu

Neil A. Ernst, John Mylopoulos
Department of Computer Science
University of Toronto
jm,nernst@cs.toronto.edu

***Abstract*—Techne is a new requirements modeling language (RML) based on the CORE ontology for requirements [1]. We motivate the need for another RML, introduce Techne through examples and review its formalization. The language supports modeling and analysis during the very early stages of requirements engineering, when the requirements problem for the system-to-be is being structured and its alternative potential solutions explored and compared in terms of how desirable they are to stakeholders, by accounting for both the relative preference of requirements and the optional or mandatory status of requirements. Its formal semantics is both paraconsistent and non-monotonic.**

***Keywords*-Requirements models, goal-oriented domain assumptions, domain modeling**

I. Introduction

Three intertwined questions remain among the very central ones in requirements engineering (RE) for at least three decades now (e.g., [2]): (1) What information should be elicited from the stakeholders of the system-to-be? (2) What models should be used to represent the elicited information? (3) What kinds of reasoning should be performed over the models of requirements? Seminal answers took the form of requirements modeling languages (RMLs), which typically included (1) an ontology of requirements to state what information to elicit and that is relevant to describe the properties and behaviors of the system-to-be and its operating environment, (2) modeling primitives corresponding to the concepts and relations of the ontology, the instances of which together form models to capture the elicited information, and (3) variously automated methods applied over the models in order to answer questions of methodological interest, such as whether a model is consistent, or if the properties and behaviors it attributes to the system allow the latter to satisfy its designated purpose. RMLs such as SADT [3], RML [4], KAOS [5] and *i** [6] often served as the starting point for further research on various issues of interest in RE, shaping thereby the field.

The answers that each RML provides to the three key questions reflect its respective fundamental assumptions about the very problem that RE aims to resolve within the broader process of systems engineering, of when that problem is resolved, and of how it can and should be resolved. SADT illustrates the views of the 1970s and 1980s, that requirements ought to be described as functions that the system-to-be should deliver once it is operational within its environment. As the importance of automated systems to the work and coordination of people increased — and interest shifted from software and hardware alone, to sociotechnical systems — it was recognized that RE must account for the variously precise and (in)consistent expectations of the stakeholders of the system-to-be, including its future users, owners, and so on, moving thereby RE away from its historical origins in formal specification methods and knowledge representation. An understanding of the functions of the system-to-be could only be sought after the beliefs, desires, and intentions of the stakeholders had been grasped to the feasible extent. The concept of system's or stakeholder's *goal* — as a means to capture desires — became central, as was made clear in KAOS. *i** went one step further with its focus for how various intentional agents/stakeholders are interdependent on each other and the system-to-be for the realization of their individual and joint goals.

The move from functions to goals reflects the change of the core ontology of RE and with it of the definition of the requirements problem. New RMLs were consequently designed, in which the modeling primitives and reasoning methods follow and benefit from the fundamental changes.

The core ontology for requirements and the requirements problem [7] — which are implicit in state-of-the-art RMLs such as KAOS and *i** — are limited in several respects that are critical for the successful performance of RE for contemporary systems [1]. In particular, it is not clear how alternative solutions to the requirements problem can be compared, e.g., what criteria (should) serve for comparison, and how these criteria are represented in requirements models. Notions of preference over requirements, and of optional and mandatory requirements are absent.

The engineering of contemporary systems requires us to address these shortcomings. A first step was a new core ontology for requirements [1], which recognizes that in addition to goals and tasks different stakeholders have different preferences over same requirements, that they are interested in choosing among alternative solutions, that potentially many alternative solutions exist (as in the case of service- and/or agent-oriented systems, in which different services or agents may compete to provide same functions), and that

requirements are not fixed, but change with new information from the stakeholders or the operating environment.

1) Contributions: We introduce *Techne*, an RML designed as a response to the challenges cited above. It is designed to assist the engineer in knowledge representation and decision making during the very early stages of requirements engineering, when the requirements problem for the system-to-be is being structured and its alternative potential solutions explored and compared in terms of how desirable they are to the stakeholders, by accounting for both the relative preference of requirements and the optional or mandatory status of requirements. A Techne requirements model captures instances of the concepts from the core ontology for requirements, allowing the representation of (i) goals, quality constraints, softgoals, domain assumptions, tasks, and (ii) relations of inference, conflict, and preference between them, along with (iii) relations to distinguish mandatory and optional requirements. Techne has formal semantics; the semantic domain is made up of structures called potential solutions of the requirements problem, which are consistent sets of requirements. Potential solutions are found via nonmonotonic and paraconsistent reasoning. It is nonmonotonic to accommodate the straightforward observation that the requirements problem and its solutions can change over time, and that conclusions drawn about these from a requirements model may not remain standing when new requirements (goals, preferences, tasks, etc.) are introduced. Reasoning is paraconsistent for an inconsistent requirements model should not allow us to conclude the satisfaction of all requirements therein.

Organization. A nontechnical introduction to requirements models and reasoning in Techne is first given via realistic examples of manageable size (§II). Syntax and semantics of the models are then presented and discussed (§III). The paper closes with a discussion of related work (§IV), a summary of conclusions, implications, and pointers for future research (§V).

II. Overview of Techne

The purpose of Techne is to support the representation and reasoning about instances of requirements problems and alternative solutions thereof. It is an RML with the three standard parts: (1) It classifies the information elicited in RE as instances of the concepts from the core ontology for requirements [1]. (2) It represents and relates these instances for a particular system-to-be in a model called the *requirements network (r-net)*, stating thereby the requirements problem for the system of interest. (3) It identifies alternative solutions and the criteria for their comparison both within a given r-net, assisting thus the subsequent choice of a solution. This section presents the intuitive ideas formalized with this RML, first in a very rough form (§II-A), then add detail (§II-B–II-C), before the subsequent section discusses the formalization (§III).

A. Intuitions

To see where and how Techne fits within the RE of a system-to-be, we need to consider what happens at the very outset of RE and go on from there. The very first step is the elicitation of requirements, the purpose of which is to acquire information about the beliefs, desires, intentions, and preferences of the system's stakeholders. Elicitation requires a double effort, one being to obtain requirements from — by interacting with — the stakeholders, the representation of these requirements being the other. The methodological aspect of how best to go about the discussions with the stakeholders is beyond the scope of Techne. Given an initial verbal, textual or otherwise account of requirements, Techne handles their *classification*, *relation* and *representation*, before and including *reasoning* thereon.

1) Classification: Techne follows the core ontology for requirements to perform the classification. Depending on their content, stakeholders' desires become goals (e.g., "Deliver music to clients via an online audio player"), quality constraints (e.g., "The bitrate of music delivered via the online audio player should be at least 128kb/s"), or softgoals (e.g., "Buffering before music starts in the audio player should be short"). Their intentions become tasks that will be accomplished either by the system-to-be, in cooperation with it, or will remain the stakeholders' responsibility. Their beliefs become domain assumptions, descriptions of conditions within which themselves and the system-to-be will be performing tasks in order to achieve the goals, quality constraints, and satisfy as best as feasible the softgoals.

2) Relation: Techne relates domain assumptions, goals, quality constraints, softgoals, and tasks via five relations. Initial requirements usually require refinement, as they tend to lack the detail necessary for, e.g., the identification of tasks the execution of which would satisfy goals (e.g., the goal "Deliver music to clients via an online audio player" may be refined onto two goals: "Music plays in a player integrated in the web page" and "Player has all standard functionalities for the listening of music", whereby the latter goal could be further refined to indicate which functionalities these are). The refinement of a goal by other goals has been a salient feature of KAOS, while other RMLs had their own proxies (e.g., task decomposition in *i**) of the refinement relation. The intuitive meaning of these relations is that if the set of more precise requirements is satisfied, then the less precise requirements are assumed satisfied. Techne considers that, say, goal refinement and task decomposition ask basically the same question: What more precise requirements should be satisfied in order to assume that the less precise — refined, decomposed — requirement is satisfied as well? Instead of relating less precise to more precise requirements by a refinement or decomposition relation, Techne generalizes these via the

inference relation which represents the application of modus ponens. One premise to the modus ponens application is the set $\{p_1, \ldots, p_n\}$ of more precise requirements. The other premise is the implication $\bigwedge_{i=1}^{n} p_i \rightarrow q$ from the conjunction of the more precise requirements to the less precise requirement, q. The conclusion is the less precise requirement q. Any inference relation captures our decision to conclude that q is satisfied if both $\{p_1, \ldots, p_n\}$ and $\bigwedge_{i=1}^{n} p_i \rightarrow q$ are, and any inference relation has this form: from $\{p_1, \ldots, p_n\}$ and $\bigwedge_{i=1}^{n} p_i \rightarrow q$, conclude q.

Techne remains in line with the purpose of the variants of the refinement relation, while it drops constraints on what instances can be refined by what other instances: some of members of $\{p_1, \ldots, p_n, q\}$ can be goals, others quality constraints, or tasks, or otherwise.[1]

Not all goals, quality constraints, tasks, or others are equally desirable (e.g., perhaps "The bitrate of music delivered via the online audio player should be at least 256kb/s" is strictly preferred to "The bitrate of music delivered via the online audio player should be at least 128kb/s"), nor can they all be satisfied together (e.g., "The bitrate of music delivered via the online audio player should be at least 256kb/s" and "Buffering before music starts in the audio player should be short"). Relative desirability is expressed via the *preference relation* in Techne. The *conflict* relation is defined over all members of a minimally inconsistent set of requirements. The final two relations are unary: the *is-mandatory relation* on a requirement indicates that the system-to-be must satisfy that requirement, while the *is-optional relation* on a requirement states that, if there were two systems, different only in that one satisfies the optional requirement and the other does not, then the former one would be strictly preferred to the second.

3) Representation: Every atomic requirement, e.g., a goal "Deliver music to clients via an online audio player", has two parts: the atomic proposition "Deliver music to clients via an online audio player" and the indication that it is a goal (i.e., an instance of the goal concept from the core ontology). The first part of an atomic requirement in Techne is represented via an atomic proposition, which itself defines a node in an r-net, while the second part is a label, assigned to the node in the r-net. Nodes for domain assumptions, goals, quality constraints, softgoals, and tasks are called *S-nodes*. The other kind of nodes, *R-nodes* represent relations between S-nodes. R-nodes are not propositions. Each of the five relations in Techne has a corresponding label, which is placed on the relevant R-node. Labeled R-nodes and S-nodes are connected via unlabeled and primitive (undefined) edges. An edge obtains its informal meaning based on which relation (i.e., R-node) it targets or originates in. E.g., if an edge originates in a goal and ends in a node for the inference relation, then the edge says that the goal is an input to the application of modus ponens. We commit to the propositional Horn clauses subset of propositional logic, hereafter PHL, which is why the inference relation captures only the application of modus ponens (and not of some other inference rule of propositional logic). It is important to keep in mind that Techne, in contrast to state-of-the-art RMLs does *not* include a visual syntax, which defines the graphical primitives used to draw r-nets. Instead, Techne has two syntaxes: a graph syntax, which is simply the representation of r-nets as labeled graphs, and a corresponding symbolic syntax, which allows us to write r-nets as sets of well-formed formulas (wffs) of Techne.

4) Reasoning: By representing the elicited requirements, the r-net for a system-to-be states the requirements problem for the system in question. The r-net is a statement of the requirements *problem* in the sense that requirements as elicited are hardly precise and detailed enough, or consistent to be taken as an end-result of RE. Firstly, they need to be refined, tasks need to be found that will satisfy the goals and not violate domain assumptions, softgoals need to be approximated by quality constraints and preferences, and so on. Secondly, elicitation is iterative, that is, requirements can change and new ones may be introduced modifying thereby the statement of the problem obtained after the initial elicitation. Thirdly, once the r-net includes conflicts, the r-net — however precise and detailed — is itself *not* the solution of the given requirements problem: the presence of conflicts means that there are alternative *potential solutions* to be found inside the r-net.

There are different kinds of solution concepts in Techne. Any conflict-free (i.e., logically consistent) part of an r-net is a *potential* solution, which is also an *admissible solution* if and only if it includes all mandatory requirements, and it is furthermore a *solution iff* it also includes at least some preferred and/or optional requirements. The preference and is-optional relations serve for the comparison of solutions. Once some solutions are found in an r-net, a *comparison table* is constructed to synthesize which solutions satisfy which optional requirements and which preferred requirements. The scope of Techne within the overall RE process stops once the comparison table is constructed. Remark that, since every consistent part of an r-net is a potential solution, it is possible to have potential solutions that are *from the methodological standpoint* (i.e., pertaining to *how* Techne is used) trivial or undesirable (e.g., an r-net equal to a single goal has that goal as its potential solution). This issue is avoided by placing additional constraints on solution concepts, and since these constraints should come from the methodology that directs the use of Techne and not Techne itself, we discuss very few of them (c.f., §III-C).

There are two salient features to the reasoning in Techne: it is *nonmonotonic* and *paraconsistent*. Nonmonotonicity comes from the simple observation that elicitation is iterative, so that requirements can be revised and new ones added. The new ones need not be consistent with those already

[1] E.g., in Techne, a goal or can be refined by other goals, tasks, and/or quality constraints. This is a departure from the standard definition of the refinement relation in RE (cf., [8]).

available: if reasoning were monotonic, solutions would not need to be revised as new requirements are added, but since new requirements may conflict with available ones, requirements in an r-net grow nonmonotonically. Solutions thus need to be recomputed as new requirements become available, which reflects the intuitive idea that initially elicited requirements are hardly ever definite, or complete. They are open to revision, so that nonmonotonic reasoning is more appropriate than its alternative. Paraconsistent reasoning is a necessity for an RML, as the other option is to admit the *ex falso quodlibet* principle: conclude anything from an inconsistency and thereby conclude any solution from an inconsistent r-net. When an r-net includes inconsistent requirements, paraconsistent reasoning guarantees that we will *not* conclude *any* solution from that r-net.

B. Modeling

We now exemplify and add details to the preceding discussion of the intuitive ideas that Techne implements.

1) Requirements Problem: Given domain assumptions, goals, quality constraints, softgoals, tasks, some of which are optional mandatory, and/or preferred over others, find tasks and domain assumptions which satisfy mandatory goals, quality constraints, and ideally also satisfy at least some of the preferred and/or optional goals and quality constraints.

2) R-Net: An r-net captures the information intervening in the requirements problem via nodes and edges in a potentially disconnected and/or cyclical and labelled graph, which represents domain assumptions, goals, quality constraints, softgoals, tasks, and relations between them. *A solution of the requirements problem captured in an r-net is itself a subgraph of that r-net which satisfies specific conditions.*

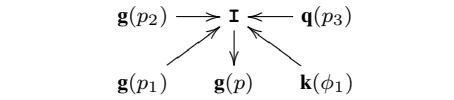
Any r-net has two kinds of nodes: *sentence/proposition (S) nodes* and *relation (R) nodes*. Any S-node is either an atomic proposition or a well-formed Horn formula (wff), labeled to indicate which concept it instantiates from the core ontology. Let p, q, r be symbols for atomic propositions and ϕ, ψ, γ for wffs, indexed or primed as needed. Any wff is either a conjunction of propositions implying another proposition (i.e., $p_1 \wedge p_2 \wedge \ldots \wedge p_n \rightarrow p$) or implying an inconsistency (i.e., $p_1 \wedge p_2 \wedge \ldots \wedge p_n \rightarrow \bot$). Following the core ontology, if an S-node refers to a condition that is believed to hold about the system-to-be and/or its relevant environment, then this S-node is an instance of the *domain assumption* concept and is labeled **k**. Desired conditions that should be satisfied are captured via instances of the *goal* (**g**), *quality constraint* (**q**), and *softgoal* (**s**) concepts. Goals describe a verifiable functional condition (e.g., "$\mathbf{g}(p_1)$: Display text ads in the audio player" in Example II.1). Quality constraints restrict the values of non-binary measurable characteristics of the system-to-be (e.g., "$\mathbf{q}(p_3)$: Maintain the player free to all users"). While a quality constraint restricts the values of qualities defined over well-defined quality spaces, a softgoal will do so over qualities with ill-defined quality spaces (e.g., "$\mathbf{s}(r)$: Sound is satisfactory"). Finally, tasks, i.e., instances of the *task* concept (**t**), capture the intentions to satisfy goals, quality constraints, and softgoals in some known manner (e.g., "$\mathbf{t}(r')$: Deliver textual ads to the audio player via a partner service").

An R-node refers to any of the following: (i) a *preference relation* between two S-nodes, (ii) a *conflict relation* between at least two S-nodes, (iii) an *inference relation* between at least two S-nodes, (iv) a unary *is-optional relation* over a single S-node, or (v) a unary *is-mandatory relation* over a single S-node. R-nodes differ from S-nodes in that an R-node is neither a proposition nor a sentence of propositional logic. S-nodes that participate in the relations are connected to respective R-nodes via the edges of the r-net. Any edge in an r-net remains unlabeled. How does then an r-net capture, say, the refinement relation? In Example II.1, the goal $\mathbf{g}(p)$ is AND-refined onto two other goals $\mathbf{g}(p_1)$ and $\mathbf{g}(p_2)$, and the quality constraint $\mathbf{q}(p_3)$. Following what we said earlier on the inference relation (cf., §II-A), if the set of the more precise requirements, i.e., $\{\mathbf{g}(p_1), \mathbf{g}(p_2), \mathbf{q}(p_3)\}$, and $\mathbf{g}(p_1) \wedge \mathbf{g}(p_2) \wedge \mathbf{q}(p_3) \rightarrow \mathbf{g}(p)$ are given, then we have a proof for $\mathbf{g}(p)$. The r-net in Example II.1 states the proof of $\mathbf{g}(p)$ from $\mathbf{g}(p_1)$, $\mathbf{g}(p_2)$, and $\mathbf{q}(p_3)$ via the application of modus ponens to the conjunction of the more precise requirements and the Horn clause (i.e., $\mathbf{g}(p_1) \wedge \mathbf{g}(p_2) \wedge \mathbf{q}(p_3) \rightarrow \mathbf{g}(p)$). An r-net captures via $\mathtt{I}$-nodes the patterns of inference that led the engineer and the stakeholders to relate the instances of the various concepts.

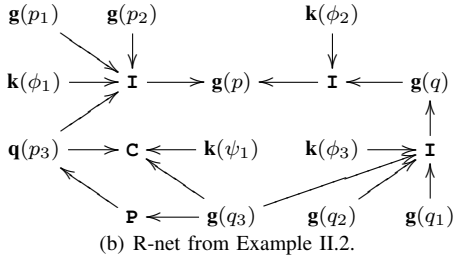
Example II.1. Suppose that the aim is to build a system that would deliver music on-demand: a user visits a website, chooses songs from a database, and can play them in the audio player on the website. Let the goal $\mathbf{g}(p)$ be the goal, where "p: Generate revenue from the audio player". Let that goal be refined by the three goals (i) "$\mathbf{g}(p_1)$: Display text ads in the audio player", (ii) "$\mathbf{g}(p_2)$: Target text ads according to users' profiles", and (iii) the quality constraint "$\mathbf{q}(p_3)$: Maintain the player free to all users". In order to derive p from the propositions in the three goals, we have $\mathbf{g}(p_1)$, $\mathbf{g}(p_2)$, and $\mathbf{q}(p_3)$, and we assume that $\mathbf{g}(p_1) \wedge \mathbf{g}(p_2) \wedge \mathbf{q}(p_3) \rightarrow \mathbf{g}(p)$. From $\{\mathbf{g}(p_1), \mathbf{g}(p_2), \mathbf{q}(p_3)\}$ and $\mathbf{g}(p_1) \wedge \mathbf{g}(p_2) \wedge \mathbf{q}(p_3) \rightarrow \mathbf{g}(p)$, we conclude $\mathbf{g}(p)$, where $\wedge$ and $\rightarrow$ are, respectively, the standard conjunction and implication connectives. Since we *assume* the implication, we say that it is a domain assumption: i.e., $\mathbf{k}(\phi_1) \equiv \mathbf{k}(\mathbf{g}(p_1) \wedge \mathbf{g}(p_2) \wedge \mathbf{q}(p_3) \rightarrow \mathbf{g}(p))$, which without labels from the r-net is saying that $\phi_1 \equiv p_1 \wedge p_2 \wedge p_3 \rightarrow p$. The r-net capturing this AND-refinement is shown in Figure 1(a).

The visual syntax of r-nets is not the topic of this paper: the graphs in Figure 1 are shown in the graph syntax of Techne. A visual syntax may be defined on top of the symbolic or graph syntax in Techne (cf., §III) to introduce graphical primitives and other features that may facilitate the construction of r-nets.■

As Example II.1 illustrates, an r-net captures a refinement by representing the proof in PHL of the refined requirement from the requirements that refine it. An r-net allows a goal to be refined by, e.g., both goals and quality constraints (as in Example II.1). Constraints, such as what concept instances can or cannot refine some concept instance (e.g., goals cannot refine a task) take the form of syntactic constraints in Techne

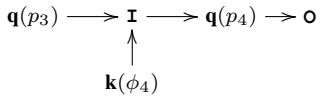


(a) R-net from Example II.1.

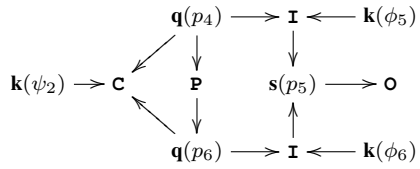


(b) R-net from Example II.2.

Figure 1. R-nets described in Examples II.1–II.2.



(a) R-net from Example II.4.



(b) R-net from Example II.5.

Figure 2. R-nets described in Examples II.4–II.5.

on the labels of S-nodes connected to an R-node. Since such concerns depend strongly on the methodology that uses Techne, few constraints are built-in (cf., §III).

Two or more S-nodes are in conflict if they cannot all appear together in a solution; a conflict R-node is labeled C. Example II.2 has the C-node C so that no solution can have $\mathbf{g}(q_3)$, $\mathbf{q}(p_3)$, and $\mathbf{k}(\mathbf{g}(q_3) \wedge \mathbf{q}(p_3) \rightarrow \bot)$. A conflict node signals that if there are solutions in the r-net, then there will be alternative solutions. It follows that these alternative solutions will need to be compared. Facing a conflict node, a preference relation can be introduced between subsets of the nodes in conflict, to indicate which of these S-nodes are strictly more desirable to have in a solution than others. Two sets of S-nodes participate in a preference relation if solutions containing one of the two sets are strictly more desirable than solutions containing the other set. A preference R-node is labeled P. In Example II.2, the preference R-node P (together with the edges entering and leaving it) that $\mathbf{g}(q_3)$ is strictly preferred to $\mathbf{q}(p_3)$.

Example II.2. (Contd. Example II.1) Suppose that "$\mathbf{g}(q)$: Charge subscription to users", and that it is AND-refined onto the conjunction of "$\mathbf{g}(q_1)$: Music database is restricted to subscribers", "$\mathbf{g}(q_2)$: Users can subscribe", and "$\mathbf{g}(q_3)$: Music player is available to subscribers only". This requires the domain assumptions $\mathbf{k}(\phi_2) \equiv \mathbf{k}(\mathbf{g}(q) \rightarrow \mathbf{g}(p))$ and $\mathbf{k}(\phi_3) \equiv \mathbf{k}(\mathbf{g}(q_1) \wedge \mathbf{g}(q_2) \wedge \mathbf{g}(q_3) \rightarrow \mathbf{g}(q))$. We also have $\mathbf{k}(\psi_1) \equiv \mathbf{k}(\mathbf{g}(q_3) \wedge \mathbf{q}(p_3) \rightarrow \bot)$, as we cannot both maintain the player free to all users ($\mathbf{q}(p_3)$) and make it available to subscribers only ($\mathbf{g}(q_3)$), so that these three cannot appear together in a solution: there is a conflict C between $\mathbf{q}(p_3)$ and $\mathbf{g}(q_3)$, when $\mathbf{k}(\psi_1)$. As there is a conflict, stakeholders will need to state which of the conflicting S-nodes they strictly prefer to the other: we will assume that it is preferred to make the music player available to subscribers only instead of making it available to all users, i.e., that $\mathbf{g}(q_3)$ is strictly preferred to $\mathbf{q}(p_3)$. If we update the r-net from Example II.1 with this information, we obtain the r-net in Figure 1(b).

The r-net in Figure 1(b) includes two AND-refinements of $\mathbf{q}(p)$. The conflict C indicates that these are two *alternative* refinements, as they cannot appear together in a solution. The r-net thus has two solutions, and the preference P says that the solution having $\mathbf{g}(q_3)$ is strictly preferred to the other that has $\mathbf{q}(p_3)$, if P is the only criterion for the comparison of the solutions. ■

When an r-net has conflicts and preferences, its evaluation will yield alternative solutions to the requirements problem. Since different sets of preferences will be satisfied in different solutions, the solutions can be compared, with each preference serving as one, among potentially many, comparison criteria.

Example II.3. (Contd. Example II.2) It is easy to see that there are two solutions in the r-net in Figure 1(b), one per AND-refinement. Let solution A be the one in which revenue is made via advertisements, and the solution B one in which subscriptions produce revenue. The solution B satisfies the preference P because it includes $\mathbf{g}(q_3)$, so that B is a more desirable solution than A. ■

Solutions are compared according to the mandatory and optional S-nodes they include, and not only according to the preference relations. Any S-node in an r-net is either (i) mandatory, when every solution must include it, (ii) optional, when it would be desirable to have that node in a solution, but not all solutions must have it, or (iii) neither mandatory nor optional. To make an S-node mandatory, we add the is-mandatory R-node labeled M to the r-net, and connect the S-node to M. If the S-node is optional, we relate it to the is-optional R-node labeled O. If an S-node is related neither to an is-mandatory, nor is-optional node, then it is assumed neither mandatory nor optional.

Example II.4. (Contd. Example II.2) If every solution must satisfy the goal $\mathbf{g}(p)$, the we can add the node M to the r-net in Example II.2, and add an edge from $\mathbf{g}(p)$ to M, indicating thereby that the goal is mandatory.

To illustrate the use of the is-optional relation, suppose that allowing access to the media player to all users ($\mathbf{q}(p_3)$) will allow new users to listen to music in an average of three clicks through the audio service ($\mathbf{q}(p_4)$) (because they do not need to register or provide their billing details). Let $\mathbf{q}(p_4)$ be optional. We consequently add the quality constraint $\mathbf{q}(p_4)$ to the r-net, along with the domain assumption $\mathbf{k}(\phi_4) \equiv \mathbf{k}(\mathbf{q}(p_3) \rightarrow \mathbf{q}(p_4))$, and the inference relation i_4 to conclude $\mathbf{q}(p_4)$ from $\mathbf{q}(p_3)$ and $\mathbf{k}(\phi_4)$. We thus added the graph in Figure 2(a) to the r-net in Example II.2.

Figure 2(a) indicates that $\mathbf{q}(p_4)$ is optional, being connected to the is-optional relation node O. If we consider the entire r-net (one in Figure 1(b) updated for the graph in Figure 2(a)), we can see that it is no longer obvious which refinement is preferred: if the solution contains $\mathbf{g}(q_3)$, then it will not contain $\mathbf{q}(p_4)$, but will have the preferred $\mathbf{g}(q_3)$; if the solution contains $\mathbf{q}(p_3)$, then it will

have $\mathbf{q}(p_4)$, but not the preferred $\mathbf{g}(q_3)$. ■

Softgoals refer to nonfunctional requirements, and must be approximated in a solution to the requirements problem. To approximate a softgoal, we seek non-softgoal S-nodes from which we can derive the softgoal.

Example II.5. (Contd. Example II.4) Suppose that we introduce the optional softgoal "$\mathbf{s}(p_5)$: It is easy for new users to access audio content" into the r-net from Example II.4. There are no universal criteria that tell us what "easy" precisely means in the context of this system-to-be. There are consequently different ways to approximate $\mathbf{s}(p_5)$. One of them consists of saying that the average number of clicks to access audio content (computed over some number of sessions and for a given focus group) is easier the fewer such clicks are needed to new users. We can consequently introduce at least two quality constraints, one being $\mathbf{q}(p_3)$ from Example II.4 and another "$\mathbf{q}(p_6)$: An average of ten clicks are needed to a new user to get to audio content", *and* a preference relation `P` to indicate that the approximation via $\mathbf{q}(p_4)$ is strictly preferred to the one via $\mathbf{q}(p_6)$. Loosely speaking, the preference tells us that the softgoal will be "more satisfied" if a solution approximates it via $\mathbf{q}(p_4)$ then via $\mathbf{q}(p_6)$. The information just given is introduced as the subgraph from Figure 2(b) to the r-net in Figure 1(b).

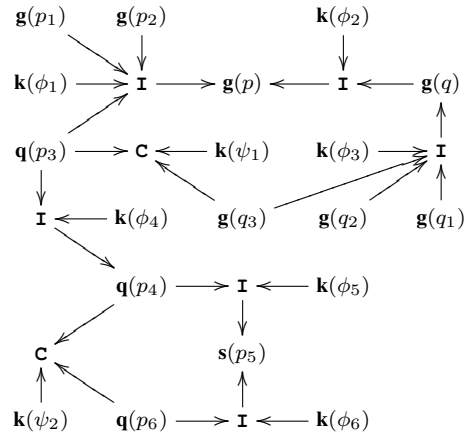
The subgraph in Figure 2(b) shows that we also need domain assumptions $\mathbf{k}(\phi_5) \equiv \mathbf{k}(\mathbf{q}(p_4) \rightarrow \mathbf{s}(p_5))$ and $\mathbf{k}(\phi_6) \equiv \mathbf{k}(\mathbf{q}(p_6) \rightarrow \mathbf{s}(p_5))$, one for each application of modus ponens, i.e., `I` and `I`, and finally, the conflict `C` to indicate that $\mathbf{k}(\psi_2) \equiv \mathbf{k}(\mathbf{q}(p_4) \wedge \mathbf{q}(p_6) \rightarrow \bot)$, and hence the approximation via $\mathbf{q}(p_4)$ is alternative to the one via $\mathbf{q}(p_6)$. ■

C. Reasoning

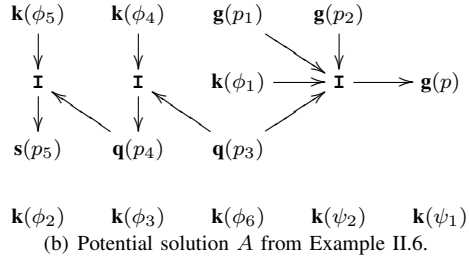
The purpose of automated reasoning facilities in Techne is to answer two questions: given an r-net, (i) What are the solutions in it? and (ii) What are the preferences and optional S-nodes that each solution contains? Example II.6 informally presents how these answers are sought in a case in which computations can be done manually.

To find solutions and compare them, i.e., to answer the two questions, we transform the r-net $\mathcal{R}$ into an *attitude-free* r-net $\bar{\mathcal{R}}$ by removing all preference, is-optional, and is-mandatory nodes, and all edges connected to these nodes. We then look for subgraphs of $\bar{\mathcal{R}}$, which are conflict-free, i.e., include no conflict nodes. We call such r-nets *potential solutions*. Among the many potential solutions, we look for the largest ones, those which, roughly speaking contain as many requirements as possible while remaining conflict-free. Such potential solutions are called *preferred potential solutions*. Given one or more preferred potential solutions, we need to establish whether each of them includes all nodes marked as mandatory — if so, then it is called an *admissible solution* for the given r-net. Given admissible solutions, we compare them by building a comparison table, which indicates for each admissible solution the optional and preferred nodes that it includes. The compared admissible solutions are simply called the *solutions* of the r-net.

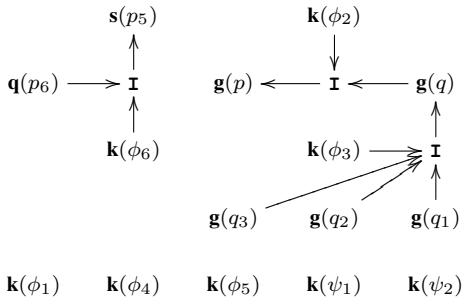
Example II.6. (Contd. Example II.5) If we remove all preference, is-optional, and is-mandatory nodes, and all edges connected to these nodes from the r-net that we have at the end of Example II.5, we obtain the attitude-free r-net in Figure 3(a).



(a) Attitude-free r-net from Example II.6.



(b) Potential solution A from Example II.6.



(c) Potential solution B from Example II.6.

Figure 3. R-nets mentioned in Example II.6.

A potential solution must be conflict-free. Figure 3(b) is an r-net representation of the potential solution $\mathcal{S}_A$, and Figure 3(c) is an r-net representation of the potential solution $\mathcal{S}_B$. We chose $\mathcal{S}_A$ and $\mathcal{S}_B$ because they are the only preferred potential solutions of the r-net, they are also admissible solutions, and finally, both of them are also solutions for the r-net of interest. Note that all domain assumptions that are also Horn clauses appear in both preferred potential solutions (i.e., $\mathbf{k}(\phi_1)$, $\mathbf{k}(\phi_2)$, $\mathbf{k}(\phi_3)$, etc.). We explain later (cf., §III-B) the reason for this, as well as why we say that Figures 3(b)–3(c) are *r-net representations of* solutions.

We can establish that the two solutions $\mathcal{S}_A$ and $\mathcal{S}_B$ have the following mandatory, optional, and preferred nodes:

- $\mathcal{S}_A$ has $\mathbf{q}(p_4)$, which `O` makes an optional node and `P` makes a preferred node; $\mathcal{S}_A$ has $\mathbf{s}(p_5)$ which `O` makes an optional node; finally, $\mathcal{S}_A$ has $\mathbf{g}(p)$ which `M` makes a mandatory node.
- $\mathcal{S}_B$ has $\mathbf{g}(q_3)$ which `P` makes a preferred node; $\mathcal{S}_B$ also has $\mathbf{s}(p_5)$ which `O` makes an optional node; finally, $\mathcal{S}_B$ has $\mathbf{g}(p)$ which `M` makes a mandatory node.

The r-net obtained at the end of Example II.5 had `P` and `P` as

preference R-nodes, **M** for its only is-mandatory R-node, and **O** and **O** for its only is-optional R-nodes. The question is: how do we compare $\mathcal{S}_A$ and $\mathcal{S}_B$? See Example II.7. ■

Given at least two solutions of an r-net, the comparison of solutions starts off with the very simple step, in which a comparison table is constructed. Each column in the table is one either **P** or **O** node. Each solution occupies a row; if the R-node in the column is in the solution, this is indicated in the table.

Example II.7. (Contd. Example II.6) The comparison table for the solutions $\mathcal{S}_A$ and $\mathcal{S}_B$ is shown below.

Solution:	**P** : $\mathbf{g}(q_3)$	**P** : $\mathbf{q}(p_4)$	**O** : $\mathbf{q}(p_4)$	**O** : $\mathbf{s}(p_5)$
$\mathcal{S}_A$	no	yes	yes	yes
$\mathcal{S}_B$	yes	no	no	yes

The table simply summarizes which of the preferences and is-optional nodes are included in which of the solutions. ■

The comparison table summarizes which solutions include which of the preferred nodes, and which of the optional nodes. The table itself does not lead us to choose one of the solutions to the requirements problem. The remaining task is to apply a decision criterion over the comparison table, in order to rank solutions. Decision criteria are beyond the scope of Techne — Techne stops after the comparison table is built.

III. Formalization

Techne is defined over propositional Horn logic (PHL): atomic propositions, clauses with no positive atom (conflicts) and ones with one positive atom (definite rules). Techne cannot be reduced to PHL alone, for at least three reasons. Firstly, both the propositions and the wffs of PHL are labeled in Techne to indicate if the proposition/wff instantiates a domain assumption, a goal, a quality constraint, a softgoal, or a task. Secondly, PHL has no proxies for preference, is-optional, and is-mandatory relations, which are used to compare the alternative solutions (i.e., alternative consistent sets of Techne wffs) to the requirements problem that an r-net states. Thirdly, Techne has a nonmonotonic and paraconsistent consequence relation $\mid\!\sim$ which is defined from the proofs in PHL that satisfy some specific constraints. These remarks are clarified in the rest of this section.

A. Graph and Symbolic Syntaxes

We have two interchangeable syntaxes for Techne. The *graph syntax* writes r-nets as graphs, and serves primarily to facilitate the discussion of the examples. The *symbolic syntax* writes r-nets as sets of Techne wffs (twffs). The two syntaxes are constructed on top of the syntactic elements of PHL, by mapping these to labels. Labeled PHL propositions and wffs are common to both syntaxes, leading us to present this shared part first below and use it to define S- and R-nodes of r-nets, and then discuss in turn the graph syntax, the symbolic syntax, and the rules for switching between them.

1) S-Nodes (Shared Syntactic Elements).: Let p, q, r be symbols that refer each to an atomic proposition, whereby an (atomic) proposition is the shareable content of an intentional state, such as a belief, desire, or an intention, i.e., what is, respectively, believed, desired, or intended. We index or prime the symbols for propositions as needed. Symbol $\bot$ reads "inconsistency".

An PHL wff is a Horn clause, and either a definite Horn clause (of the form $\bigwedge_{i=1}^{n} p_i \rightarrow q$) or a compound negation ($\bigwedge_{i=1}^{n} p_i \rightarrow \bot$). The following BNF grammar defines wffs in PHL, for $n \geq 1$:

$$\textit{wff} ::= \quad \bigwedge_{i=1}^{n} p_i \rightarrow q \mid \bigwedge_{i=1}^{n} p_i \rightarrow \bot$$

While propositions and wffs of PHL do reflect the relations between the elements/agents of the system-to-be and of its operating environment, i.e., the conditions in which these may stand, they alone are of limited interest, for they abstract from the pragmatic component: neither the propositions nor the wffs say whether the relations they describe are desired, believed, or otherwise. That pragmatic component is essential in RE because it distinguishes between a proposition/wff instantiating a goal or a domain assumption, or otherwise, and thereby influencing the kinds of analyses/transformations applicable to it. The core ontology chooses the pragmatics for propositions/wffs based on the intentional state inferred from the way in which the proposition/wff was communicated, and defines corresponding concepts of domain assumption, functional goal, quality constraint, softgoal, and task [1].

Every PHL proposition and wff becomes a member of the set S of S-nodes of an r-net after it is labeled to indicate which concept it instantiates. Labeling rules fully define the labeling function for S-nodes.

Definition III.1. ***Labeling Function for S-nodes ($\mathcal{L}_S$).*** *Let $\mathcal{L}_S : S \longrightarrow L_S$, where $L_S = \{\boldsymbol{k}, \boldsymbol{g}, \boldsymbol{q}, \boldsymbol{s}, \boldsymbol{t}\}$.*

- *$\mathcal{L}_S(p) = \boldsymbol{k}$ (resp. $\mathcal{L}_S(\phi) = \boldsymbol{k}$) iff p (resp. ϕ) instantiates a domain assumption;*
- *$\mathcal{L}_S(p) = \boldsymbol{g}$ iff p instantiates a functional goal;*
- *$\mathcal{L}_S(p) = \boldsymbol{q}$ iff p instantiates a quality constraint;*
- *$\mathcal{L}_S(p) = \boldsymbol{s}$ iff p instantiates a softgoal; and*
- *$\mathcal{L}_S(p) = \boldsymbol{t}$ iff p instantiates a task.*

Remark III.2. We abbreviate $(p, \mathcal{L}_S(p) = \mathbf{x})$ by $\mathbf{x}(p)$, where $\mathbf{x} \in L_S$, and $(\phi, \mathcal{L}_S(\phi) = \mathbf{k})$ by $\mathbf{k}(\phi)$.

Domain assumptions have a special role in Techne: when an S-node is a labeled PHL wff, we assume that this labeled wff is as a whole an instance of a domain assumption. This corresponds to the intuition that when we say, e.g., $\mathbf{g}(p) \wedge \mathbf{g}(q) \rightarrow \mathbf{g}(r)$ that we believe that the conjunction of $\mathbf{g}(p)$ and $\mathbf{g}(q)$ implies $\mathbf{g}(r)$, so that $\mathbf{g}(p) \wedge \mathbf{g}(q) \rightarrow \mathbf{g}(r)$ is an instance of the domain assumption concept.

Any S-node, denoted *sn*, is generated via the BNF grammar in Equations 1–3 obtained by labeling PHL propositions and wffs with $\mathcal{L}_S$.

$$pl ::= \quad \mathbf{k}(p) \mid \mathbf{g}(p) \mid \mathbf{q}(p) \mid \mathbf{s}(p) \mid \mathbf{t}(p) \tag{1}$$

$$\phi ::= \quad \bigwedge_{i=1}^{n} pl_i \rightarrow pl \mid \bigwedge_{i=1}^{n} pl_i \rightarrow \bot \tag{2}$$

$$sn ::= \quad pl \mid \mathbf{k}(\phi) \tag{3}$$

Rules 1–3 guarantee that an S-node is either a labeled proposition or a Horn domain assumption. The latter, $\mathbf{k}(\phi)$ is a *Horn* domain assumption as it is either a definite Horn clause (when $\phi = \bigwedge_{i=1}^{n} pl_i \rightarrow pl$) or a compound negation (when $\phi = \bigwedge_{i=1}^{n} pl_i \rightarrow \bot$), as in Example II.2.

2) R-Nodes (Shared Syntactic Elements): R-nodes capture relations between requirements referred to by S-nodes. The set R of R-nodes in an r-net is partitioned as follows: (i) $R_{\mathtt{I}}$ is the set of *inference* nodes, (ii) $R_{\mathtt{C}}$ the set of *conflict* nodes, (iii) $R_{\mathtt{P}}$ the set of *preference* nodes, (iv) $R_{\mathtt{O}}$ the set of *is-optional* nodes, and (v) $R_{\mathtt{M}}$ the set of *is-mandatory* nodes. Members of the partitions are labeled by the labeling function $\mathcal{L}_R : R \longrightarrow L_R$.

Definition III.3. ***Inference Relation.*** *For every $S_j \subseteq S$ such that $S_j = \{pl_i \mid 1 \leq i \leq n\} \cup \{\mathbf{k}\,(\bigwedge_{i=1}^{n} pl_i \rightarrow pl)\} \cup \{pl\}$ and $n \geq 1$, there is an inference relation in the r-net between members of $\{pl_i \mid 1 \leq i \leq n\} \cup \{\mathbf{k}\,(\bigwedge_{i=1}^{n} pl_i \rightarrow pl)\}$ and pl, symbolized by an R-node $rn_j \in R$ labeled $\mathcal{L}_R(rn_j) = \mathtt{I}$.*

An inference node refers to the application of the modus ponens (MP) inference rule to a nonempty set of its input S-nodes, called *premises* in order to derive another output S-node, called the *conclusion*. The inference node is present in an r-net whenever the premises are present, as MP applies to derive the conclusion S-node.

Definition III.4. ***Conflict Relation.*** *For every $S_j \subseteq S$ such that $S_j = \{pl_i \mid 1 \leq i \leq n\} \cup \{\mathbf{k}\,(\bigwedge_{i=1}^{n} pl_i \rightarrow \bot)\}$ and $n \geq 2$, there is a conflict relation in the r-net between members of S_j, symbolized by an R-node $rn_j \in R$ labeled $\mathcal{L}_R(rn_j) = \mathtt{C}$.*

A conflict relation stands between elements of a minimally inconsistent set of S-nodes. The conflict relation indicates that $\bot$ was concluded from the application of MP, in which the premise is the set $\{pl_i \mid 1 \leq i \leq n\} \cup \{\mathbf{k}\,(\bigwedge_{i=1}^{n} pl_i \rightarrow \bot)\}$.

While the inference and conflict relations are defined as applications of inference rules to S-nodes, the preference, is-mandatory, and is-optional relations have no corresponding notion in PHL.

Definition III.5. ***Preference Relation.*** *For every two nonempty sets $S_i, S_j \subseteq S$ there is a preference relation in the r-net between these two sets stating that S_i is strictly preferred to S_j, symbolized by the R-node $rn_k \in R$ labeled $\mathcal{L}_R(rn_k) = \mathtt{P}$, iff:*

1) *$S_i \cap S_j = \emptyset$, and*
2) *whenever two solutions $\mathcal{S}_1$ and $\mathcal{S}_2$ for the given r-net differ only in that $\mathcal{S}_1$ has all members of S_i but none of S_j while $\mathcal{S}_2$ has all members of S_j but none of S_i, the solution $\mathcal{S}_1$ is strictly more desirable to the solution $\mathcal{S}_2$.*

Definition III.6. ***Is-Mandatory Relation.*** *There is a unary is-mandatory relation on an S-node $pl \in S$ in the r-net, symbolized by the R-node $rn_j \in R$ labeled $\mathcal{L}_R(rn_j) = \mathtt{M}$ iff pl must appear in every potential solution of the requirements problem defined by that r-net.*

Definition III.7. ***Is-Optional Relation.*** *There is a unary is-optional relation on an S-node $pl \in S$ in the r-net, symbolized by the R-node $rn_j \in R$ labeled $\mathcal{L}_R(rn_j) = \mathtt{O}$ iff whenever two solutions $\mathcal{S}_1$ and $\mathcal{S}_2$ differ only in that $\mathcal{S}_1$ has pl and $\mathcal{S}_2$ does not have pl, then $\mathcal{S}_1$ is strictly more desirable to $\mathcal{S}_2$.*

Remark III.8. $\mathbf{y}_i$ abbreviates $(rn_i \in R, \mathcal{L}_R(rn_i) = \mathbf{y})$, where $\mathbf{y} \in L_R$, since the *rn* symbol stays for any $\mathbf{y}$ and any i.

An r-net, denoted $\mathcal{R}$ is simply a structure that has both S-nodes and R-nodes, as they have been defined above.

Definition III.9. ***R-Net ($\mathcal{R}$).*** *An r-net is the tuple $\mathcal{R} = (S, R)$, where S is the set of S-nodes and R is the set of R-nodes over members of S.*

3) Graph Syntax: An r-net $\mathcal{R}$ is in graph syntax a directed, labeled, and potentially disconnected and/or cyclical graph $G(\mathcal{R}) = (N, E, \iota)$, where $N = S \cup R$ is the set of nodes, $E \subseteq N \times N$ are directed edges, and $\iota : E \longrightarrow N \times N$ is the incidence function that maps every edge to its origin and destination node. Every S-node and R-node is a node of $G(\mathcal{R})$ while the edges serve to connect the relata of an R-node to that R-node.

Remark III.10. As a convention, we say that an R-node *rn* has a nonempty set of inputs, denoted *in*(*rn*) and a potentially empty set of outputs, *out*(*rn*), where: (a) $in(rn) = \{sn \mid (sn, rn) \in E\}$ is the set of all S-nodes that are connected to *rn* via edges that target *rn* in the given r-net, and (b) $out(rn) = \{sn \mid (rn, sn) \in E\}$ is the set of all S-nodes that are connected to *rn* via edges that originate in *rn*.

To ensure that a $G(\mathcal{R})$ is syntactically valid, its incidence function ι must connect its nodes so that exactly the following constraints are satisfied:

1) for every $\mathtt{I} \in R$ relating the members of $\{pl_i \mid 1 \leq i \leq n\} \cup \{\mathbf{k}\,(\bigwedge_{i=1}^{n} pl_i \rightarrow pl)\}$ to *pl*: $in(\mathtt{I}) = \{pl_i \mid 1 \leq i \leq n\} \cup \{\mathbf{k}\,(\bigwedge_{i=1}^{n} pl_i \rightarrow pl)\}$ and $out(\mathtt{I}) = pl$;
2) for every $\mathtt{C} \in R$ relating the members of $\{pl_i \mid 1 \leq i \leq n\} \cup \{\mathbf{k}\,(\bigwedge_{i=1}^{n} pl_i \rightarrow \bot)\}$: $in(\mathtt{C}) = \{pl_i \mid 1 \leq i \leq n\} \cup \{\mathbf{k}\,(\bigwedge_{i=1}^{n} pl_i \rightarrow \bot)\}$ and $out(\mathtt{C} = \emptyset$;
3) for every $\mathtt{P} \in R$ relating $S_1 \subseteq S$ to $S_2 \subseteq S$: $in(p) = S_1$ and $out(p) = S_2$;
4) for every $\mathtt{M} \in R$ relating *pl* to itself: $in(\mathtt{M}) = pl$ and $out(\mathtt{M}) = \emptyset$;
5) for every $\mathtt{O} \in R$ relating *pl* to itself: $in(\mathtt{O}) = pl$ and $out(\mathtt{O}) = \emptyset$.

We assume that any $G(\mathcal{R})$ r-net mentioned in this paper is syntactically valid.

4) Symbolic Syntax: R-nets or fragments thereof can be written via the symbolic syntax of Techne, the wffs of which are simply inline rewritings of nodes and/or edges, in which the connective $\twoheadrightarrow$ stands for an edge in the r-net. In symbolic syntax an r-net is a set of Techne wffs (twffs), $\mathcal{R} = \{twff_1, \ldots, twff_m\},\ m \geq 1$. Twffs are generated via the following BNF grammar, where *sn* is an S-node (cf., Equation 3) and *rn* is an R-node:

$$\begin{aligned}
iwff ::= \quad & (pl_1, \ldots, pl_n, \mathbf{k}(\bigwedge_{i=1}^{n} pl_i \rightarrow pl)) \twoheadrightarrow \mathtt{I}(rn) \\
& \mid \mathtt{I}(rn) \twoheadrightarrow pl && (4)\\
cwff ::= \quad & (nl_1, \ldots, nl_n, \mathbf{k}(\bigwedge_{i=1}^{n} pl_i \rightarrow \bot)) \twoheadrightarrow \mathtt{C}(rn) && (5)\\
pwff ::= \quad & (pl_1, \ldots, pl_n) \twoheadrightarrow \mathtt{P}(rn) \\
& \mid \mathtt{P}(rn) \twoheadrightarrow (pl_1, \ldots, pl_n) && (6)\\
mwff ::= \quad & pl \twoheadrightarrow \mathtt{M}(rn) && (7)\\
owff ::= \quad & pl \twoheadrightarrow \mathtt{O}(rn) && (8)\\
twff ::= \quad & sn \mid pl_1, \ldots, pl_n \mid iwff \mid cwff \mid pwff \\
& \mid mwff \mid owff && (9)
\end{aligned}$$

Equations 1–9 define the symbolic syntax of r-nets in Techne. It is not difficult to see that an r-net defined as a set of tffws is a syntactically valid r-net. Remark that *iwff*, *cwff*, *pwff*, *mwff*, and *owff* are due to, respectively, Definitions III.3, III.4, III.5, III.6, and III.7.

Example III.11. Figure 1(a) in symbolic syntax is: $\mathcal{R} = \{\mathbf{g}(p_1), \mathbf{g}(p_2), \mathbf{q}(p_3), \mathbf{k}(\phi_1), \mathtt{I}, \mathbf{g}(p), (\mathbf{g}(p_1), \mathbf{g}(p_2), \mathbf{q}(p_3), \mathbf{k}(\phi_1)) \twoheadrightarrow \mathtt{I}, \mathtt{I} \twoheadrightarrow \mathbf{g}(p)\}$. ■

5) Conversion between the Graph and Symbolic Syntax: Equations 10–15 define correspondences between the graph and symbolic syntaxes, for an r-net $\mathcal{R}$ the graph of which has the set S of S-nodes, set R of R-nodes, and set E of edges. Elements of symbolic syntax are on left-hand side of "≡", those of graph syntax are on the right in each of the Equations 10–15.

$$sn \in S \equiv sn \in S \quad (10)$$

$$rn \in R \equiv rn \in R \quad (11)$$

$$(sn \twoheadrightarrow rn) \in \mathcal{R} \equiv (sn, rn) \in E,\ sn \in S,\ rn \in R \quad (12)$$

$$(rn \twoheadrightarrow sn) \in \mathcal{R} \equiv (rn, sn) \in E,\ sn \in S,\ rn \in R \quad (13)$$

$$((sn_1, \ldots, sn_n) \twoheadrightarrow rn) \in \mathcal{R} \equiv$$
$$\forall i,\ 1 \leq i \leq n,\ (sn_i, rn) \in E,\ sn_i \in S,\ rn \in R \quad (14)$$

$$(rn \twoheadrightarrow (sn_1, \ldots, sn_n)) \in \mathcal{R} \equiv$$
$$\forall i,\ 1 \leq i \leq n,\ (rn, sn_i) \in E,\ sn_i \in S,\ rn \in R \quad (15)$$

Remark III.12. The symbolic syntax lets us write an r-net as a set of twwfs, so that we can write $\mathcal{R}_2 \subset \mathcal{R}_1$ to say that $\mathcal{R}_2$ was obtained by removing nodes and/or edges from $\mathcal{R}_1$, or in terms of graph syntax, that $\mathcal{R}_2$ is a subgraph of $\mathcal{R}_1$.

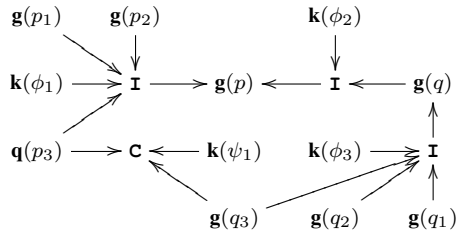


Figure 4. An attitude-free r-net, $\bar{\mathcal{R}}$.

B. Microsolutions

An r-net in its whole states the requirements problem for a given system-to-be, and includes all potential solutions to that problem. Microsolutions are the building blocks of potential solutions, which in turn serve to define the semantics of r-nets and the solution concepts in Techne. Microsolutions are sought in a subgraph of $\mathcal{R}$, called the *attitude-free* r-net $\bar{\mathcal{R}}$ which lacks preferences, is-optional, and is-mandatory nodes from the original $\mathcal{R}$.

Definition III.13. ***Attitude-Free R-Net.*** *The attitude-free r-net $\bar{\mathcal{R}}$ of $\mathcal{R}$ is obtained from $\mathcal{R}$ by removing all preference, is-mandatory, and is-optional relations from $\mathcal{R}$.*

Remark III.14. It is obvious that $\bar{\mathcal{R}} \subseteq \mathcal{R}$ and that every r-net has a unique attitude-free variant. Also, $G(\bar{\mathcal{R}})$ is obtained from $G(\mathcal{R})$ by eliminating all **P**-, **M**-, and **O**-nodes and edges entering or leaving these nodes in $G(\mathcal{R})$.

Example III.15. Figure 4 shows the attitude-free r-net $\bar{\mathcal{R}}$ obtained by removing preference, is-mandatory, and is-optional relations from the r-net $\mathcal{R}$ in Figure 1(b). ■

An inference node that concludes an *pl* is a step in a proof for that *pl* in $\mathcal{R}$. When the premises to an **I**-node are themselves conclusions of other **I**-nodes, we can identify proofs for an *pl* in $\mathcal{R}$. A proof for *pl* is a subgraph of $\mathcal{R}$, which we call a *microsolution for pl*. The reason we wish to identify microsolutions is that we combine them to build potential solutions. In order to identify microsolutions in an r-net, we first need the consequence relation $\vdash_\tau$.

Definition III.16. ***Consequence Relation*** $\vdash_\tau$**.** *Given $\bar{\mathcal{R}} = (\bar{S}, \bar{R})$, $\bar{S}' \subseteq \bar{S}$, and $x \in \{pl, \bot\}$:*

1) *$\bar{S}' \vdash_\tau pl$ if $pl \in \bar{S}'$, or*
2) *$\bar{S}' \vdash_\tau x$ if $\forall 1 \leq i \leq n,\ \bar{S}' \vdash_\tau pl_i$ and $\mathbf{k}(\bigwedge_{i=1}^{n} pl_i \rightarrow x) \in \bar{S}'$.*

Proposition III.17. *$\vdash_\tau$ is paraconsistent.*

Proof: (By contradiction.) $\vdash_\tau$ is paraconsistent iff $\forall pl_i \neq pl_j \neq pl_k,\ \{pl_i, pl_j, \mathbf{k}(pl_i \wedge pl_j \rightarrow \bot)\} \nvdash_\tau pl_k$. Suppose that $\vdash_\tau$ is not paraconsistent, so that $\forall pl_i \neq pl_j \neq pl_k,\ \{pl_i, pl_j, \mathbf{k}(pl_i \wedge pl_j \rightarrow \bot)\} \vdash_\tau pl_k$. $\{pl_i, pl_j, \mathbf{k}(pl_i \wedge pl_j \rightarrow \bot)\} \vdash_\tau pl_k$ contradicts the first condition in Definition III.16, as $pl_k \notin \{pl_i, pl_j, \mathbf{k}(pl_i \wedge pl_j \rightarrow \bot)\}$. It also contradicts the second condition in Definition III.16 as there is no Horn clause with pl_k as its positive literal in $\{pl_i, pl_j, \mathbf{k}(pl_i \wedge pl_j \rightarrow \bot)\}$. ■

Proposition III.17 confirms that we cannot conclude anything from a contradiction in Techne. Also note that our $\vdash_\tau$ is sound w.r.t. standard entailment in propositional logic, but is incomplete in two ways: it only considers deducing positive atoms, and no ordinary proofs based on arguing by contradiction go thru, thus being paraconsistent.

We will be building potential solutions out of microsolutions, fragments of $\bar{\mathcal{R}}$ defined via $\vdash_\tau$. In the definition of microsolution, we use the set H, in which we place all Horn domain assumptions, each of which is either of the form $\mathbf{k}(\bigwedge_{i=1}^{n} pl_i \rightarrow pl)$ or $\mathbf{k}(\bigwedge_{i=1}^{n} pl_i \rightarrow \bot)$. We require that H is consistent, which leads us to introduce correct r-nets.

Definition III.18. ***Correct Attitude-free R-Net.*** *An $\bar{\mathcal{R}}$ is correct iff $H \nvdash_\tau \bot$, where $H = \{\boldsymbol{k}(\phi) \mid \boldsymbol{k}(\phi) \in \bar{\mathcal{R}}\}$.*

We assume that all attitude-free r-nets mentioned in this paper are correct. It is obvious that if $\bar{\mathcal{R}}$ is correct, then so is $\mathcal{R}$ since H is identical in $\bar{\mathcal{R}}$ and $\mathcal{R}$.

Remark III.19. We denote *Source*$(\mathcal{R})$ the set of all source nodes in $\mathcal{R}$: *Source*$(\mathcal{R}) = \{sn \mid sn \in S,\ in(sn) = \emptyset\}$.

Definition III.20. ***Microsolution.*** *An $\bar{S}'$ is a microsolution for $pl \in \bar{S}$ in $\bar{\mathcal{R}}$, denoted $\langle \bar{S}', pl \rangle_{\bar{\mathcal{R}}}$, iff:*

1) $\bar{S}' \vdash_\tau pl$,
2) $\bar{S}' \subseteq$ *Source*$(\bar{\mathcal{R}})$,
3) $\bar{S}' \nvdash_\tau \bot$,
4) $H \subset \bar{S}'$, *where* $H = \{\boldsymbol{k}(\phi) \mid \boldsymbol{k}(\phi) \in \bar{\mathcal{R}}\}$,
5) $\bar{S}'$ *is minimal, i.e.,* $\nexists \bar{S}'' \subset \bar{S}'$ *such that* $\bar{S}'' \vdash_\tau pl$ *and* $H \subset \bar{S}''$.

The first condition in Definition III.20 requires that there is a derivation of *pl* from $\bar{S}'$. The second condition requires that all members of $\bar{S}'$ are source S-nodes in the $\bar{\mathcal{R}}$. The third condition requires that $\bar{S}'$ is consistent. H in the fourth condition is the set of all Horn domain assumptions in $\bar{\mathcal{R}}$, and is used to indicate that every microsolution must include all Horn domain assumptions. The fourth condition is necessary because we build potential solutions from microsolutions, by putting together microsolutions which are consistent: if the fourth condition is missing, it would be possible to build potential solutions simply by eliminating Horn domain assumptions which indicate conflict, i.e., those of the form $\mathbf{k}(\bigwedge_{i=1}^{n} pl_i \rightarrow \bot)$, which would be erroneous since the resulting potential solutions would simply ignore conflicts (i.e., as if there were no conflicts in $\bar{\mathcal{R}}$, or equivalently as if $\bar{\mathcal{R}}$ was consistent in the first place). The fifth condition indicates that $\bar{S}'$ is minimal, containing on top of H only those S-nodes which are necessary and sufficient for the derivation of the conclusion of the microsolution.

Remark III.21. When it is clear from the text that $\langle \bar{S}', sn \rangle_{\bar{\mathcal{R}}}$ is a microsolution in $\bar{\mathcal{R}}$, we omit the subscript and write $\langle \bar{S}', sn \rangle$.

Example III.22. The following are all microsolutions found in $\bar{\mathcal{R}}$ that is shown in Figure 4:

- For $\bar{\mathcal{R}}$, $H = \{\mathbf{k}(\phi_1), \mathbf{k}(\phi_2), \mathbf{k}(\phi_3), \mathbf{k}(\psi_1)\}$, so that $\langle H \cup \{\mathbf{g}(p_1)\}, \mathbf{g}(p_1)\rangle$, $\langle H \cup \{\mathbf{g}(p_2)\}, \mathbf{g}(p_2)\rangle$, $\langle H \cup \{\mathbf{q}(p_3)\}, \mathbf{q}(p_3)\rangle$, $\langle H \cup \{\mathbf{g}(q_1)\}, \mathbf{g}(q_1)\rangle$, $\langle H \cup \{\mathbf{g}(q_2)\}, \mathbf{g}(q_2)\rangle$, $\langle H \cup \{\mathbf{g}(q_3)\}, \mathbf{g}(q_3)\rangle$. Note that all conclusions and premises of these micrsolutions are source nodes in $\bar{\mathcal{R}}$ and all S-nodes in these microsolutions are thus in *Source*$(\bar{\mathcal{R}})$.
- $\langle H \cup \{\mathbf{g}(p_1), \mathbf{g}(p_2), \mathbf{q}(p_3)\}, \mathbf{g}(p)\rangle$. This microsolution is one of the two refinements of $\mathbf{g}(p)$ in $\bar{\mathcal{R}}$.
- $\langle H \cup \{\mathbf{g}(q_1), \mathbf{g}(q_2), \mathbf{g}(q_3)\}, \mathbf{g}(p)\rangle$. This microsolution is the second refinement of $\mathbf{g}(p)$ in $\bar{\mathcal{R}}$.
- $\langle H \cup \{\mathbf{g}(q_1), \mathbf{g}(q_2), \mathbf{g}(q_3)\}, \mathbf{g}(q)\rangle$. This microsolution is part of the other refinement of $\mathbf{g}(p)$.

Remark that no microsolution concludes any of the members of H, as Definition III.20 states that the conclusion of a microsolution can only be a *pl* and Equations 1–3 distinguish *pl* from Horn domain assumptions. ■

There can be different microsolutions for the same *pl*, and that this depends on the presence/absence of different MP applications (i.e., `I`-nodes) which conclude *pl* in an $\bar{\mathcal{R}}$. Stated otherwise, if there are several proofs for *pl* in $\bar{\mathcal{R}}$, then there are several microsolutions for *pl* in $\bar{\mathcal{R}}$. A microsolution for an *pl* is the smallest subgraph of an $\bar{\mathcal{R}}$ that contains only requirements which are in favor of *pl*. A microsolution for *pl* does not contain *all* requirements that are in favor of *pl*, since there can be more than one microsolution for *pl* in $\bar{\mathcal{R}}$.

C. Semantics

The semantic domain of an r-net is made up of structures called *solutions*. There are several kinds of solutions, the simplest among which is the *potential solution*. Any potential solution is simply a conflict-free set of microsolutions, i.e., a set of microsolutions in which the premises of all microsolutions in it are consistent.

Definition III.23. ***Potential Solution.*** *A set $\mathcal{S}$ of microsolutions from $\bar{\mathcal{R}}$ is called a potential solution for $\bar{\mathcal{R}}$ iff $\mathcal{S} = \{\langle \bar{S}_i, pl_i \rangle \mid \bigcup_{i=1}^{n} \bar{S}_i \nvdash_\tau \bot\}$.*

Example III.24. The following are all potential solutions of the $\bar{\mathcal{R}}$ in Figure 4:

- Recall that for $\bar{\mathcal{R}}$, $H = \{\mathbf{k}(\phi_1), \mathbf{k}(\phi_2), \mathbf{k}(\phi_3), \mathbf{k}(\psi_1)\}$ so that each of these singletons is a potential solution: $\langle H \cup \{\mathbf{g}(p_1)\}, \mathbf{g}(p_1)\rangle$, $\langle H \cup \{\mathbf{g}(p_2)\}, \mathbf{g}(p_2)\rangle$, $\langle H \cup \{\mathbf{q}(p_3)\}, \mathbf{q}(p_3)\rangle$, $\langle H \cup \{\mathbf{g}(q_1)\}, \mathbf{g}(q_1)\rangle$, $\langle H \cup \{\mathbf{g}(q_2)\}, \mathbf{g}(q_2)\rangle$, $\langle H \cup \{\mathbf{g}(q_3)\}, \mathbf{g}(q_3)\rangle$.
- Let $\mathcal{S}$ be a set of microsolutions such that (i) $\mathcal{S}$ is a set of singleton microsolutions and (ii) $\{\langle H \cup \{\mathbf{q}(p_3)\}, \mathbf{q}(p_3)\rangle, \langle H \cup \{\mathbf{g}(q_3)\}, \mathbf{g}(q_3)\rangle\} \not\subset \mathcal{S}$. Every $\mathcal{S}$ is a potential solution.
- Let $\mathcal{S}_1 = \{\langle H \cup \{\mathbf{g}(p_1)\}, \mathbf{g}(p_1)\rangle, \langle H \cup \{\mathbf{g}(p_2)\}, \mathbf{g}(p_2)\rangle, \langle H \cup \{\mathbf{q}(p_3)\}, \mathbf{q}(p_3)\rangle, \langle H \cup \{\mathbf{g}(p_1), \mathbf{g}(p_2), \mathbf{q}(p_3)\}, \mathbf{g}(p)\rangle\}$, so that $\mathcal{S}_1$ is a potential solution and every member of $\wp\mathcal{S}_1$ is a potential solution as well, where $\wp\mathcal{S}_1$ is a powerset of $\mathcal{S}_1$.
- Let $\mathcal{S}_2 = \{\langle H \cup \{\mathbf{g}(q_1)\}, \mathbf{g}(q_1)\rangle, \langle H \cup \{\mathbf{g}(q_2)\}, \mathbf{g}(q_2)\rangle, \langle H \cup \{\mathbf{g}(q_3)\}, \mathbf{g}(q_3)\rangle, \langle H \cup \{\mathbf{g}(q_1), \mathbf{g}(q_2), \mathbf{g}(q_3)\}, \mathbf{g}(q)\rangle, \langle H \cup \{\mathbf{g}(q_1), \mathbf{g}(q_2), \mathbf{g}(q_3)\}, \mathbf{g}(p)\rangle\}$, so that $\mathcal{S}_2$ is a potential solution and every member of $\wp\mathcal{S}_2$ is a potential solution as well.

Remark that any set of microsolutions that has both $\langle H \cup \{\mathbf{q}(p_3)\}, \mathbf{q}(p_3)\rangle$ and $\langle H \cup \{\mathbf{g}(q_3)\}, \mathbf{g}(q_3)\rangle$ cannot be a potential solution because $H \cup \{\mathbf{q}(p_3), \mathbf{g}(q_3)\} \vdash_\tau \bot$. ■

Given an $\bar{\mathcal{R}}$, every member pl of $Source(\bar{\mathcal{R}})$ gives a microsolution $\langle pl, pl\rangle$, so that every nonempty $\bar{\mathcal{R}}$ has at least one potential solution. Not all potential solutions are equally interesting: we want to find the largest potential solutions in an $\bar{\mathcal{R}}$. To do that, we can start with a potential solution $\mathcal{S}$ and add microsolutions to it. The question is, given $\mathcal{S}$, which microsolution can we add to $\mathcal{S}$ while still keeping it a potential solution? We can add microsolutions to a potential solutions as long as doing so does not result in inconsistency. We consequently have a preferred potential solution as a potential solution that is maximal w.r.t. set inclusion.

Definition III.25. ***Preferred Potential Solution.*** *If $\mathcal{S}$ is a potential solution of $\bar{\mathcal{R}}$, then $\mathcal{S}$ is also a preferred potential solution for $\bar{\mathcal{R}}$ iff there is no other potential solution $\mathcal{S}'$ of $\bar{\mathcal{R}}$ such that $\mathcal{S} \subset \mathcal{S}'$.*

Example III.26. The $\bar{\mathcal{R}}$ in Figure 4 has two preferred potential solutions:

- $\mathcal{S}_1 = \{\langle H \cup \{\mathbf{g}(p_1)\}, \mathbf{g}(p_1)\rangle, \langle H \cup \{\mathbf{g}(p_2)\}, \mathbf{g}(p_2)\rangle, \langle H \cup \{\mathbf{q}(p_3)\}, \mathbf{q}(p_3)\rangle, \langle H \cup \{\mathbf{g}(p_1), \mathbf{g}(p_2), \mathbf{q}(p_3)\}, \mathbf{g}(p)\rangle\}$.
- $\mathcal{S}_2 = \{\langle H \cup \{\mathbf{g}(q_1)\}, \mathbf{g}(q_1)\rangle, \langle H \cup \{\mathbf{g}(q_2)\}, \mathbf{g}(q_2)\rangle, \langle H \cup \{\mathbf{g}(q_3)\}, \mathbf{g}(q_3)\rangle, \langle H \cup \{\mathbf{g}(q_1), \mathbf{g}(q_2), \mathbf{g}(q_3)\}, \mathbf{g}(q)\rangle, \langle H \cup \{\mathbf{g}(q_1), \mathbf{g}(q_2), \mathbf{g}(q_3)\}, \mathbf{g}(p)\rangle\}$.

Remark that each preferred potential solution has all the elements of the respective refinements of $\mathbf{g}(p)$, which were discussed in Examples II.1 and II.2

As Example III.26 illustrates, an $\bar{\mathcal{R}}$ can have one *or more* preferred potential solutions. The question now is which of the preferred potential solutions gives an *admissible solution*. To find the answer, we check which (if any) of the preferred potential solutions contain all mandatory S-nodes. Recall that we obtained $\bar{\mathcal{R}}$ from $\mathcal{R}$, so that answering this question requires that we take into account the is-mandatory relations from $\mathcal{R}$.

Definition III.27. ***Admissible Solution.*** *A set $\mathcal{S}$ of microsolutions from $\bar{\mathcal{R}}$ is an admissible solution for $\mathcal{R}$ iff:*

1) *$\mathcal{S}$ is a preferred potential solution of $\bar{\mathcal{R}}$;*
2) *$\forall pl_i \in \mathcal{R}$ such that $\exists \boldsymbol{m}_j \in \mathcal{R}$, $pl_i = in(\boldsymbol{m}_j)$, there is a microsolution $\langle \cdot, pl_i\rangle$ in $\mathcal{S}$, i.e., there is for every mandatory S-node pl_i from $\mathcal{R}$ a microsolution for that mandatory S-node pl_i in $\mathcal{S}$.*

Example III.28. Suppose that $\mathbf{g}(p)$ is the only mandatory S-node in the r-net shown in Figure 4, then there are two admissible solutions for that r-net:

- $\mathcal{S}_1 = \{\langle H \cup \{\mathbf{g}(p_1)\}, \mathbf{g}(p_1)\rangle, \langle H \cup \{\mathbf{g}(p_2)\}, \mathbf{g}(p_2)\rangle, \langle H \cup \{\mathbf{q}(p_3)\}, \mathbf{q}(p_3)\rangle, \langle H \cup \{\mathbf{g}(p_1), \mathbf{g}(p_2), \mathbf{q}(p_3)\}, \mathbf{g}(p)\rangle\}$.
- $\mathcal{S}_2 = \{\langle H \cup \{\mathbf{g}(q_1)\}, \mathbf{g}(q_1)\rangle, \langle H \cup \{\mathbf{g}(q_2)\}, \mathbf{g}(q_2)\rangle, \langle H \cup \{\mathbf{g}(q_3)\}, \mathbf{g}(q_3)\rangle, \langle H \cup \{\mathbf{g}(q_1), \mathbf{g}(q_2), \mathbf{g}(q_3)\}, \mathbf{g}(q)\rangle, \langle H \cup \{\mathbf{g}(q_1), \mathbf{g}(q_2), \mathbf{g}(q_3)\}, \mathbf{g}(p)\rangle\}$.

The above are also the only two preferred potential solutions for the r-net in Figure 4.

Suppose that both $\mathbf{g}(p)$ and $\mathbf{g}(q)$ are mandatory in the r-net in Figure 4. In that case, the r-net has only one admissible solution, which is $\mathcal{S}_2$ above. ■

A solution of $\mathcal{R}$ is defined as an admissible solution that has microsolutions for at least some optional and/or preferred S-nodes in $\mathcal{R}$.

Definition III.29. ***Solution.*** *A set $\mathcal{S}$ of microsolutions from $\bar{\mathcal{R}}$ is an admissible solution for $\mathcal{R}$ iff:*

1) *$\mathcal{S}$ is an admissible solution for $\mathcal{R}$;*
2) *at least one of the conditions below holds:*
   a) *for an $pl_i \in \mathcal{R}$ such that $\exists \boldsymbol{o}_j \in \mathcal{R}$, $pl_i = in(\boldsymbol{o}_j)$, there is a microsolution $\langle \cdot, pl_i\rangle$ in $\mathcal{S}$, i.e., there is for an optional S-node pl_i from $\mathcal{R}$ a microsolution for that optional S-node pl_i in $\mathcal{S}$;*
   b) *for $m \geq 1$ and some S-nodes $\{pl_1, \ldots, pl_m\} \subset \mathcal{R}$ such that $\exists \boldsymbol{p}_j \in \mathcal{R}$, $\{pl_1, \ldots, pl_m\} = in(\boldsymbol{p}_j)$, there is in $\mathcal{S}$ a microsolution $\langle \cdot, pl_i\rangle$ for every $1 \leq i \leq m$, i.e., for preferred S-nodes $\{pl_1, \ldots, pl_m\}$ from $\mathcal{R}$ there are microsolutions for each of these preferred nodes in $\mathcal{S}$.*

Example III.30. Suppose that in the r-net in Figure 1(b), $\mathbf{g}(p)$ is the only mandatory S-node, so that both $\mathcal{S}_1$ and $\mathcal{S}_2$ from Example III.28 are admissible solutions for that r-net. We see in Figure 1(b) that there are no optional S-nodes and that $\mathbf{g}(q_3)$ is preferred to $\mathbf{q}(p_3)$, so that $\mathcal{S}_2$ is a solution (as it has a microsolution for $\mathbf{g}(q_3)$) for that r-net and $\mathcal{S}_1$ is not a solution for the same r-net.

Following the discussions above, we can define the nonmonotonic $\mathrel{|\!\sim}$ consequence relation.

Definition III.31. ***Consequence Relation $\mathrel{|\!\sim}$.*** *Given an $\bar{\mathcal{R}}$, an S-node pl in $\bar{\mathcal{R}}$ is a $\mathrel{|\!\sim}$-consequence of a set X of S-nodes of $\bar{\mathcal{R}}$, denoted $X \mathrel{|\!\sim} pl$ iff there is a microsolution $\langle X', pl\rangle$ s.t. $X' \subseteq X$.*

The set of admissible solutions does not grow monotonically as an r-net increases, which reflects the intuition that requirements given early on in the process are frequently revised, so that no requirement is ever definite and stable, but tentative and open to revision. E.g., let $\bar{\mathcal{R}}$ have $Source(\bar{\mathcal{R}}) = \{pl_1, \mathbf{k}(pl_1 \rightarrow pl_2)\}$ so that there is a microsolution for pl_2. Further, let $\bar{\mathcal{R}}'$ have $Source(\bar{\mathcal{R}}') = \{pl_1, \mathbf{k}(pl_1 \rightarrow pl_2), \mathbf{k}(pl_1 \wedge pl_2 \rightarrow \bot)\}$. While $\bar{\mathcal{R}}$ had a microsolution for pl_2, $\bar{\mathcal{R}}'$ which has all requirements from $\bar{\mathcal{R}}$ along with other requirements does not have a microsolution for pl_2. If the set of admissible solutions were to grow monotonically, then the admissible solutions we found in $\bar{\mathcal{R}}$ would remain among the admissible solutions of $\bar{\mathcal{R}}'$. We arguably adopt a more realistic stance in Techne, according to which conflict S-nodes may be added to $\bar{\mathcal{R}}$, so that there is no guarantee that the admissible solutions from $\bar{\mathcal{R}}$ would remain the admissible solutions of $\bar{\mathcal{R}}'$.

According to the core ontology for requirements [1], the requirements problem is this: Given domain assumptions, goals, quality constraints, softgoals, tasks, and preferences — some of which are optional or mandatory — in, respectively **K**, **G**, **Q**, **S**, **T**, and **P**, find parts $\mathbf{K}^*, \mathbf{T}^*, \mathbf{G}^*, \mathbf{Q}^*, \mathbf{P}^*$ of these structures, such that: $\mathbf{K}^*, \mathbf{T}^* \mathrel{|\!\sim} \mathbf{G}^*, \mathbf{Q}^*, \mathbf{P}^*$, where $\mathrel{|\!\sim}$ is a nonmonotonic consequence relation, whereby the softgoals (**S**)

are missing beacuse they are all assumed to be approximated. How does Techne rewrite the requirements problem? In Techne, an attitude-free r-net $\bar{\mathcal{R}}$ has domain assumptions (i.e., a set **K** of domain assumption S-nodes), tasks (**T**), goals (**G**), quality constraints (**Q**), and softgoals (**S**), so that we have the information required by the problem formulation. We also have preferences, is-optional, and is-mandatory relations, but we leave them outside $\bar{\mathcal{R}}$ and in the corresponding $\mathcal{R}$. Once we identify solutions in Techne, we have not yet solved the requirements problem. We need to take two remaining steps. The first one is as follows: among all solutions we identified, consider only those in which every source node is either a domain assumption (i.e., in **K**) or a task (i.e., in **T**). This condition intuitively means that we are taking only solutions in which the source S-nodes do not need to be refined or operationalized (i.e., if a goal is a source node, we need to operationalize it, by identifying tasks the execution of which satisfies the goal). The second, and final step is to compare these solutions in the comparison table, and apply a decision rule in order to rank them based on preferences and is-optional relations. It follows that in Techne, a solution to the requirements problem is a Techne solution in which every source node is either a domain assumption or a task. Finally, it is emphasized in the definition of the requirements problem that $\mid\!\sim$ is a non-monotonic consequence relation, which Techne reflects by having the set of admissible solutions grow non-monotonically.

D. Dialectical Semantics

Remark III.32. For some given arbitrary set of microsolutions $\mathcal{S} = \{\langle \bar{S}_1, pl_1\rangle, \ldots, \langle \bar{S}_n, pl_n\rangle\}$, we call $\mathit{Base}(\mathcal{S}) = \bigcup_{i=1}^{n} \bar{S}_i$ the base of $\mathcal{S}$. $\mathit{Base}(\mathcal{S})$ can be inconsistent. ■

The semantic domain of an r-net is made up of solutions, all variants of which are a specialization of the potential solution concept. According to Definition III.23, a set of microsolutions $\mathcal{S} = \{\langle \bar{S}_i, pl_i\rangle \mid 1 \leq i \leq n\}$ is a potential solution iff its base is consistent, i.e., iff $\mathit{Base}(\mathcal{S}) \not\vdash_{\tau} \bot$. Since every set of microsolutions with a consistent base is a potential solution, we were interested in preferred potential solutions: Definition III.25 says that every potential solution maximal w.r.t. $\subseteq$ is a preferred potential solution. Recall that the admissible solution and solution concepts are basically preferred potential solutions which satisfy some additional properties pertaining to preference, is-mandatory, and is-optional relations. Since the admissible solution and solution concepts are defined on top of the potential solution and preferred potential solution concepts, our discussion below will remain interested in attitude-free r-nets, in which we are dealing with potential solutions and their preferred variants only.

The intuitive idea elaborated in the rest of this section is that there is another solution concept in Techne, called *robust potential solution*, where "robust" has its usual sense: something is robust if it can withstand stresses, pressures, or changes in procedure or circumstance. Roughly speaking, a potential solution will be robust if it includes information that "defends" that potential solution against information which is outside of it and disputes the information in the potential solution. The disputing information may include, e.g., domain assumptions about exceptions in which some goals in the potential solution will fail to be satisfied, or tasks which are not in the potential solution, but which can be performed in the environment of the system-to-be, and which, if they are executed, will block some task in the potential solution to be executed. If a potential solution defends itself, it protects itself from the disputing information outside of it, and we can thus say that that potential solution is robust.

We turn to examples first before formalizing robustness. Example III.33 introduces the idea of robustness via a generic example which has the benefit of being very simple and yet highlight the principal intuitions. Example III.34 illustrates these intuitions in a slightly more complex case, in which we discuss the robustness of the r-net in Example II.1 (cf., Figure 1(a)).

Example III.33. Suppose that we have an $\bar{\mathcal{R}}$ which only has a single S-node, $\mathbf{x}(p)$. We will keep this a generic example, so that we do not care which proposition p refers to, and it is unimportant what concept p instantiates (it can be a goal, a task, a softgoal, etc. — i.e., $\mathbf{x} \in L_S$). Since $\bar{\mathcal{R}}$ has only $\mathbf{x}(p)$:

- $\bar{\mathcal{R}}$ has no Horn domain assumptions, i.e., $H = \emptyset$;
- $\bar{\mathcal{R}}$ has a single microsolution, $\langle \mathbf{x}(p), \mathbf{x}(p)\rangle$;
- $\bar{\mathcal{R}}$ has only one potential solution $\mathcal{S} = \{\langle \mathbf{x}(p), \mathbf{x}(p)\rangle\}$;
- $\bar{\mathcal{R}}$ has only one preferred potential solution, $\mathcal{S}$.

$\mathcal{S}$ is a robust potential solution of $\bar{\mathcal{R}}$, clearly, not only because it is the only potential solution of $\bar{\mathcal{R}}$ *but also* because there is nothing in $\bar{\mathcal{R}}$ that is in conflict with $\mathcal{S}$.

We now add the following two S-nodes to $\bar{\mathcal{R}}$:

- $\mathbf{y}(q)$, where q refers to some proposition, and $\mathbf{y} \in L_S$ (**y** can, but need not be same as **x**);
- $\mathbf{k}(\phi) \equiv \mathbf{k}(\mathbf{x}(p) \wedge \mathbf{y}(q) \rightarrow \bot)$, which is a domain assumption stating that $\mathbf{x}(p)$ and $\mathbf{y}(q)$ are in conflict.

By adding these two S-nodes to $\bar{\mathcal{R}}$, we obtained $\bar{\mathcal{R}}'$. In terms of graph syntax, $G(\bar{\mathcal{R}}')$ is as shown below:

$$\mathbf{x}(p) \longrightarrow \mathtt{C} \longleftarrow \mathbf{y}(q)$$

We can say the following about $\bar{\mathcal{R}}'$:

- $\bar{\mathcal{R}}'$ has one Horn domain assumption, i.e., $H' = \{\mathbf{k}(\phi)\}$;
- $\bar{\mathcal{R}}'$ has two microsolutions: $\langle H' \cup \{\mathbf{x}(p)\}, \mathbf{x}(p)\rangle$ and $\langle H' \cup \{\mathbf{y}(q)\}, \mathbf{y}(q)\rangle$;
- $\bar{\mathcal{R}}'$ has two potential solutions: $\mathcal{S}_p = \{\langle H' \cup \{\mathbf{x}(p)\}, \mathbf{x}(p)\rangle\}$ and $\mathcal{S}_q = \{\langle H' \cup \{\mathbf{y}(q)\}, \mathbf{y}(q)\rangle\}$;
- $\mathcal{S}_p$ and $\mathcal{S}_q$ are the only two preferred potential solutions of $\bar{\mathcal{R}}'$.

It is important to observe above that the only difference between $\mathcal{S}$ and $\mathcal{S}_p$ is that the microsolution $\langle \mathbf{x}(p), \mathbf{x}(p)\rangle$ in $\mathcal{S}$ "became" the microsolution $\langle H' \cup \{\mathbf{x}(p)\}, \mathbf{x}(p)\rangle$ only by adding H' to it.

Intuitively, we had the potential solution $\mathcal{S}$ above, and then "found out" (came upon, discovered) $\mathbf{y}(q)$ and that $\mathbf{y}(q)$ and $\mathbf{x}(p)$ cannot hold together, as $\mathbf{k}(\phi)$ tells us. We thus end up in $\bar{\mathcal{R}}'$ with two preferred potential solutions. Now, we could say that $\mathbf{x}(p)$ is preferred to $\mathbf{y}(q)$. Preference of $\mathbf{x}(p)$ over $\mathbf{y}(q)$ would have us choose $\mathcal{S}_p$ over $\mathcal{S}_q$, yet that would be an uncomfortable choice,

since it is one that ignores the presence of the information (i.e., $\mathbf{y}(q)$ and $\mathbf{k}(\phi)$) that disputes the chosen option $\mathcal{S}_q$. If we do choose $\mathcal{S}_p$, we have chosen a (preferred) potential solution which is not robust. To choose instead a robust option we can keep the preference of $\mathbf{x}(p)$ over $\mathbf{y}(q)$ and make robust the potential solution that has $\mathbf{x}(p)$ by adding information that disputes the disputing information; we can do that by adding the following two S-nodes to $\bar{\mathcal{R}}'$:

- $\mathbf{z}(r)$, where r refers to some proposition, and $\mathbf{z} \in L_S$ ($\mathbf{z}$ can, but need not be same as any of $\mathbf{x}$ or $\mathbf{z}$);
- $\mathbf{k}(\psi) \equiv \mathbf{k}(\mathbf{y}(q) \wedge \mathbf{z}(r) \rightarrow \bot)$, which is a domain assumption stating that $\mathbf{y}(q)$ and $\mathbf{z}(r)$ are in conflict.

By adding these two S-nodes to $\bar{\mathcal{R}}'$, we obtain $\bar{\mathcal{R}}''$. In terms of graph syntax, $G(\bar{\mathcal{R}}'')$ is as shown below:

$$\mathbf{x}(p) \longrightarrow \mathtt{C} \longleftarrow \mathbf{y}(q) \longrightarrow \mathtt{C} \longleftarrow \mathbf{z}(r)$$

We can say the following about $\bar{\mathcal{R}}''$:

- $\bar{\mathcal{R}}''$ has two Horn domain assumptions, i.e., $H'' = \{\mathbf{k}(\phi), \mathbf{k}(\psi)\}$;
- $\bar{\mathcal{R}}''$ has three microsolutions:
  - $\langle H'' \cup \{\mathbf{x}(p)\}, \mathbf{x}(p)\rangle$,
  - $\langle H'' \cup \{\mathbf{y}(q)\}, \mathbf{y}(q)\rangle$,
  - $\langle H'' \cup \{\mathbf{z}(r)\}, \mathbf{z}(r)\rangle$,
- $\bar{\mathcal{R}}''$ has four two potential solutions:
  - $\mathcal{S}_p = \{\langle H'' \cup \{\mathbf{x}(p)\}, \mathbf{x}(p)\rangle\}$,
  - $\mathcal{S}_q = \{\langle H'' \cup \{\mathbf{y}(q)\}, \mathbf{y}(q)\rangle\}$,
  - $\mathcal{S}_r = \{\langle H'' \cup \{\mathbf{z}(r)\}, \mathbf{z}(r)\rangle\}$,
  - $\mathcal{S}_{pr} = \mathcal{S}_p \cup \mathcal{S}_r$.
- $\mathcal{S}_q$ and $\mathcal{S}_{pr}$ are the only two preferred potential solutions of $\bar{\mathcal{R}}''$.

$\mathcal{S}_{pr}$ is reinforced, and robust in the sense that we have added information to $\bar{\mathcal{R}}'$ that ends up being consistent with $\mathbf{x}(p)$ and inconsistent with $\mathbf{y}(q)$. We say that $\mathcal{S}_{pr}$ is robust w.r.t. $\mathbf{y}(q)$. ■

Example III.34. Figure 1(a) shows the r-net presented in Example II.1. Observe that that r-net $\mathcal{R}$ has no preference, is-optional, and is-mandatory relations, so that $\mathcal{R} = \bar{\mathcal{R}}$. Moverover, it is easy to see that:

- $H = \{\mathbf{k}(\phi_1)\}$;
- there are four microsolutions in $\bar{\mathcal{R}}$: $\mathcal{S} = \{\langle H \cup \{\mathbf{g}(p_1)\}, \mathbf{g}(p_1)\rangle, \langle H \cup \{\mathbf{g}(p_2)\}, \mathbf{g}(p_2)\rangle, \langle H \cup \{\mathbf{q}(p_3)\}, \mathbf{q}(p_3)\rangle, \langle H \cup \{\mathbf{g}(p_1), \mathbf{g}(p_2), \mathbf{q}(p_3)\}, \mathbf{g}(p)\rangle\}$;
- every member of $\wp\mathcal{S}$ is a potential solution.

Suppose that we have the following two domain assumptions:

- "$\mathbf{k}(r)$: For comparable music delivery systems, bandwidth cost grows faster then advertising revenue.",
- "$\mathbf{k}(\psi_3) \equiv \mathbf{k}(\mathbf{k}(r) \wedge \mathbf{q}(p_3) \wedge \mathbf{g}(p) \rightarrow \bot)$: It is impossible to generate revenue from the audio player ($\mathbf{g}(p)$) and maintain the player free to all users ($\mathbf{q}(q_3)$) while for comparable systems bandwidth cost grows faster then advertising revenue ($\mathbf{k}(r)$)".

What happened above is that we added new information $\mathbf{k}(r)$ and $\mathbf{k}(\psi_3)$ to $\bar{\mathcal{R}}$, obtaining thereby another attitude-free r-net $\bar{\mathcal{R}}'$, in which there is a conflict between $\mathbf{g}(p)$, $\mathbf{q}(q_3)$, and $\mathbf{k}(r)$. Given $\bar{\mathcal{R}}'$, it is not difficult to establish that:

- $H' = \{\mathbf{k}(\phi_1), \mathbf{k}(\psi_3)\}$;
- microsolutions in $\bar{\mathcal{R}}$ are: $\mathcal{S}' = \{\langle H' \cup \{\mathbf{g}(p_1)\}, \mathbf{g}(p_1)\rangle, \langle H' \cup \{\mathbf{g}(p_2)\}, \mathbf{g}(p_2)\rangle, \langle H' \cup \{\mathbf{q}(p_3)\}, \mathbf{q}(p_3)\rangle, \langle H' \cup \{\mathbf{g}(p_1), \mathbf{g}(p_2), \mathbf{q}(p_3)\}, \mathbf{g}(p)\rangle\}, \langle H' \cup \{\mathbf{k}(r)\}, \mathbf{k}(r)\rangle\}$;
- all members of $\wp\mathcal{S}'$ *which do not include* $\boldsymbol{g}(p)$, $\boldsymbol{q}(q_3)$, *and* $\boldsymbol{k}(r)$ *together* are potential solutions of $\bar{\mathcal{R}}'$;
- $\wp\mathcal{S} \subset \wp\mathcal{S}'$ so that all potential solutions of $\bar{\mathcal{R}}$ are also potential solutions of $\bar{\mathcal{R}}'$ — stated otherwise, the set of potential solutions grows monotonically (we will discuss this observation in more detail later).

It is critical to understand what happened when we moved from $\bar{\mathcal{R}}$ to $\bar{\mathcal{R}}'$. Firstly, observe that we obtained $\bar{\mathcal{R}}'$ by *adding* information to $\bar{\mathcal{R}}$; we removed nothing from $\bar{\mathcal{R}}$. Secondly, recall that $\bar{\mathcal{R}}$ showed an AND-refinement of $\mathbf{g}(p)$, and since it had no conflicts, it showed a preferred potential solution for $\mathbf{g}(p)$: $\mathcal{S}$ is that preferred potential solution. Thirdly, $\mathcal{S}$ is *also* a preferred potential solution of $\bar{\mathcal{R}}'$ *because we obtained* $\bar{\mathcal{R}}'$ *by adding only information which generates a conflict with parts of* $\mathcal{S}$. Now, what that new information does is that it disputes $\mathcal{S}$, in that the only thing this new information says is that it is impossible to generate revenue from the audio player ($\mathbf{g}(p)$) and maintain the player free to all users ($\mathbf{q}(q_3)$) while for comparable systems bandwidth cost grows faster then advertising revenue ($\mathbf{k}(r)$).

What happened when we moved from $\bar{\mathcal{R}}$ to $\bar{\mathcal{R}}'$ is that we added to $\bar{\mathcal{R}}$ information which disputes the preferred potential solution $\mathcal{S}$ which we found in $\bar{\mathcal{R}}$. Now, it is clear that $\mathcal{S}$ is a preferred potential solution of $\bar{\mathcal{R}}'$ as well, but the interesting question is would we be satisfied with $\mathcal{S}$ given $\bar{\mathcal{R}}'$? We *could*, of course, since $\mathcal{S}$ indeed is a preferred potential solution of $\bar{\mathcal{R}}'$, but doing so goes against a very basic observation, namely, that *if we are content with* $\mathcal{S}$ *given* $\bar{\mathcal{R}}'$, *we are ignoring that there is information that disputes* $\mathcal{S}$. Stated otherwise, $\mathcal{S}$ is not robust.

To obtain a *robust* preferred potential solution, say, $\mathcal{S}''$, we need to make sure that $\mathcal{S}''$ "defends itself" from the disputing information. We do this by strengthening $\mathcal{S}$, and one way we can strengthen it is by adding the following information to $\bar{\mathcal{R}}'$, which results in $\bar{\mathcal{R}}''$:

- "$\mathbf{g}(r_1)$: Reduce bandwidth costs."
- "$\mathbf{g}(r_2)$: Sign partnership agreements with bandwidth owners."
- "$\mathbf{k}(\psi_5) \equiv \mathbf{k}(\mathbf{g}(r_2) \rightarrow \mathbf{g}(r_1))$: Partnerships with bandwidth owners reduce bandwidth costs."
- "$\mathbf{k}(\psi_4) \equiv \mathbf{k}(\mathbf{g}(r_1) \wedge \mathbf{k}(r) \rightarrow \bot)$: It is impossible for bandwidth cost to grow faster then advertising revenue when bandwidth cost is reduced."

If we add the four S-nodes above to $\bar{\mathcal{R}}'$, we obtain $\bar{\mathcal{R}}''$, in which:

- $H'' = \{\mathbf{k}(\phi_1), \mathbf{k}(\psi_3), \mathbf{k}(\psi_4), \mathbf{k}(\psi_5)\}$;
- $\mathcal{S}'' = \{\langle H'' \cup \{\mathbf{g}(p_1)\}, \mathbf{g}(p_1)\rangle, \langle H'' \cup \{\mathbf{g}(p_2)\}, \mathbf{g}(p_2)\rangle, \langle H'' \cup \{\mathbf{q}(p_3)\}, \mathbf{q}(p_3)\rangle, \langle H'' \cup \{\mathbf{g}(p_1), \mathbf{g}(p_2), \mathbf{q}(p_3)\}, \mathbf{g}(p)\rangle\}, \langle H'' \cup \{\mathbf{k}(r)\}, \mathbf{k}(r)\rangle, \langle H'' \cup \{\mathbf{g}(r_2)\}, \mathbf{g}(r_2)\rangle, \langle H'' \cup \{\mathbf{g}(r_2)\}, \mathbf{g}(r_1)\rangle$.

$\mathcal{S}$ is a potential solution of $\bar{\mathcal{R}}''$, but is not a preferred potential solution of it, since we added some information that does not conflict with $\mathcal{S}$, namely $\mathbf{g}(r_1)$, $\mathbf{g}(r_2)$ and $\mathbf{k}(\psi_5)$. Instead, a preferred potential solution is: $\mathcal{S}_x = \mathcal{S}'' \backslash \{\langle H'' \cup \{\mathbf{k}(r)\}, \mathbf{k}(r)\rangle\}$. What makes $\mathcal{S}_x$ interesting is that it now includes the initial AND-refinement of $\mathbf{g}(p)$ *and also* the goal to reduce bandwith costs $\mathbf{g}(r_1)$ via partnership agreements $\mathbf{g}(r_2)$, which goes against $\mathbf{k}(r)$ via $\mathbf{k}(\psi_4)$.

We started thus from a potential solution, learned that under some conditions ($\mathbf{k}(r)$) it is not robust, and made it robust by requiring that an additional goal be satisfied ($\mathbf{g}(r_2)$). We went from a potential solution $\mathcal{S}$ that was not robust to one that is, $\mathcal{S}_x$. ■

Once we can compare potential solutions of a given $\bar{\mathcal{R}}$ in terms of robustness, we are no longer interested in preferred potential solutions of $\bar{\mathcal{R}}$, but in those which we can make or which are robust. Moreover, not all potential solutions are equally robust, and just how robust a potential solution is will depend on which disputing information it can defend itself from.

The question of interest at this point is: How to find robust potential solutions in some given $\bar{\mathcal{R}}$? Roughly speaking, the idea is to define a relation between consistent sets of

potential solutions, called the attack relation, and then look at how potential solutions defend itself from attacks of other potential solution, whereby how robust a potential solution is will depend on "how well" it defends itself. We proceed to explain in detail and formalize this idea in several steps below. Firstly (§III-D1), we introduce the *attack* relation between consistent sets of potential solutions (and in some special cases, between individual potential solutions, or between individual microsolutions). We then show that consistent sets of potential solutions together with the attack relation between them can be understood as an abstract argumentation framework, so that any $\bar{\mathcal{R}}$ can be redefined as an abstract argumentation framework (§III-D2). This link to abstract argumentation frameworks leads us to define what a robust potential solution is, and explain how to build one from a potential solution (§III-D3). We finally show that dialectical reasoning plays a fundamental role in the modeling or requirements by relating robust potential solutions to Dung's extensions of abstract argumentation frameworks [9] (§III-D4).

1) Attack Relation: Potential solutions attack each other when the union of their bases is inconsistent. It follows that the attack relation is not a primitive, but reflects the presence of inconsistency. This begs the question of why we need the attack relation at all, since we already have the conflict relation. Convenience in the presentation of results below is a secondary motivation, the principal one being that the notion of attack plays a central role in abstract argumentation, so that having an attack relation lets us establish more straighfrowardly the link between r-nets and abstract argumentation frameworks.

Definition III.35. ***Attack Relation.*** *Two consistent sets of potential solutions,* $X = \{\mathcal{S}_1, \ldots, \mathcal{S}_n \mid \bigcup_{i=1}^{n} \mathit{Base}(\mathcal{S}_i) \not\vdash_\tau \bot\}$ *and* $Y = \{\mathcal{S}_{n+1}, \ldots, \mathcal{S}_{n+m} \mid \bigcup_{i=n+1}^{n+m} \mathit{Base}(\mathcal{S}_i) \not\vdash_\tau \bot\}$ *attack each other, denoted* $X \xrightarrow{A} Y$*, iff* $\bigcup_{i=1}^{n+m} \mathit{Base}(\mathcal{S}_i) \vdash_\tau \bot$*, for* $n \geq 1$ *and* $m \geq 1$.

Remark III.36. Suppose that $\{\mathcal{S}_1, \ldots, \mathcal{S}_n \mid \bigcup_{i=1}^{n} \mathit{Base}(\mathcal{S}_i) \not\vdash_\tau \bot\}$ and $\{\mathcal{S}_{n+1}, \ldots, \mathcal{S}_{n+m} \mid \bigcup_{i=n+1}^{n+m} \mathit{Base}(\mathcal{S}_i) \not\vdash_\tau \bot\}$ attack each other. We can say that potential solutions attack each other, or that microsolutions attack each other in the following cases:

- if $n = m = 1$, then we have two potential solutions $\mathcal{S}_1$ and $\mathcal{S}_2$ which attack each other, $\mathcal{S}_1 \xrightarrow{A} \mathcal{S}_2$ and :
  - also $\mathcal{S}_2 \xrightarrow{A} \mathcal{S}_1$, i.e., attack is symmetric;
  - at least some of the microsolutions in $\mathcal{S}_1$ are not acceptable w.r.t. $\mathcal{S}_2$, and *vice-versa*, i.e., at least some members of $\mathcal{S}_2$ are not acceptable w.r.t. $\mathcal{S}_1$;
  - $\mathcal{S}_1 \cup \mathcal{S}_1$ is not a potential solution.
- if $n = m = 1$ and $|\mathcal{S}_1| = |\mathcal{S}_2| = 1$, then two microsolutions (one in each of $\mathcal{S}_1$ and $\mathcal{S}_2$) attack each other, i.e., $\langle \bar{S}_1, pl_1 \rangle \xrightarrow{A} \langle \bar{S}_2, pl_2 \rangle$.

The remarks above illustrate that attack is a relation that holds between different structures that are internally consistent — i.e., between microsolutions, potential solutions, or consistent sets of potential solutions — but produce inconsistency when put together. ■

Remark III.37. The attack relation between *two potential solutions*, e.g., $\mathcal{S}_A \xrightarrow{A} \mathcal{S}_B$ is specialized as follows:

- An attack is called a *many-to-many* attack of $\mathcal{S}_A$ on $\mathcal{S}_B$ iff $\exists \mathcal{S}'_A \subseteq \mathcal{S}_A,\ \mathcal{S}'_B \subseteq \mathcal{S}_B$ such that $\mathit{Base}(\mathcal{S}'_A) \cup \mathit{Base}(\mathcal{S}'_B) \vdash_\tau \bot$ and $|\mathcal{S}'_A| > 1$ and $|\mathcal{S}'_B| > 1$.
- An attack is called a *one-to-many attack* of $\mathcal{S}_A$ on $\mathcal{S}_B$ iff $\exists \mathcal{S}'_A \subseteq \mathcal{S}_A,\ \mathcal{S}'_B \subseteq \mathcal{S}_B$ such that $\mathit{Base}(\mathcal{S}'_A) \cup \mathit{Base}(\mathcal{S}'_B) \vdash_\tau \bot$ and $|\mathcal{S}'_A| = 1$ and $|\mathcal{S}'_B| > 1$.
- An attack is called a *many-to-one attack* of $\mathcal{S}_A$ on $\mathcal{S}_B$ iff $\exists \mathcal{S}'_A \subseteq \mathcal{S}_A,\ \mathcal{S}'_B \subseteq \mathcal{S}_B$ such that $\mathit{Base}(\mathcal{S}'_A) \cup \mathit{Base}(\mathcal{S}'_B) \vdash_\tau \bot$ and $|\mathcal{S}'_A| > 1$ and $|\mathcal{S}'_B| = 1$.
- An attack is called a *one-to-one attack* of $\mathcal{S}_A$ on $\mathcal{S}_B$ iff $\exists \mathcal{S}'_A \subseteq \mathcal{S}_A,\ \mathcal{S}'_B \subseteq \mathcal{S}_B$ such that $\mathit{Base}(\mathcal{S}'_A) \cup \mathit{Base}(\mathcal{S}'_B) \vdash_\tau \bot$ and $|\mathcal{S}'_A| = 1$ and $|\mathcal{S}'_B| = 1$.

The one-to-one and many-to-one attack relations can be further specialized by looking into the microsolutions that participate in the attack:

- An (one-to-one or many-to-one) attack of $\mathcal{S}_A$ on $\mathcal{S}_B = \{\langle \bar{S}_j, pl_j \rangle\}$ is also a rebutal iff $\mathit{Base}(\mathcal{S}'_A) \cup \{pl_j\} \vdash_\tau \bot$.
- An (one-to-one or many-to-one) attack of $\mathcal{S}_A$ on $\mathcal{S}_B = \{\langle \bar{S}_j, pl_j \rangle\}$ is also an undercut iff $\mathit{Base}(\mathcal{S}'_A) \cup \{pl_j\} \not\vdash \bot$ and $\mathit{Base}(\mathcal{S}'_A) \cup \bar{S}_j \vdash_\tau \bot$. ■

Example III.38. Several simple examples of attack relations are given below:

- Many-to-one rebutal: Suppose that we have three microsolutions $\langle H \cup \{pl_1\}, pl_1 \rangle$, $\langle H \cup \{pl_2\}, pl_2 \rangle$, and $\langle H \cup \{pl_3\}, pl_3 \rangle$, and that $H = \{\mathbf{k}(pl_1 \wedge pl_2 \wedge pl_3 \rightarrow \bot)\}$. The three microsolutions cannot appear together in a potential solution, but any pair or single one of them can. E.g., $\mathcal{S} = \{\langle H \cup \{pl_1\}, pl_1 \rangle, \langle H \cup \{pl_2\}, pl_2 \rangle\}$ attacks $\langle H \cup \{pl_3\}, pl_3 \rangle$, because $H \cup \{pl_1, pl_2\} \cup \{pl_3\} \vdash_\tau \bot$, i.e., there is a many-to-one rebutal on $\langle H \cup \{pl_3\}, pl_3 \rangle$ from $\mathcal{S}$ by $\langle H \cup \{pl_1\}, pl_1 \rangle$ and $\langle H \cup \{pl_2\}, pl_2 \rangle$.
- Many-to-one undercut: Suppose that we have three microsolutions $\langle H \cup \{pl_1\}, pl_1 \rangle$, $\langle H \cup \{pl_2\}, pl_2 \rangle$, and $\langle H \cup \{pl_3\}, pl_4 \rangle$, and that $H = \{\mathbf{k}(pl_1 \wedge pl_2 \wedge pl_3 \rightarrow \bot), \mathbf{k}(pl_3 \rightarrow pl_4)\}$. The three microsolutions cannot appear together in a potential solution, but any pair or single one of them can. There is a many-to-one undercut attack on $\langle H \cup \{pl_3\}, pl_4 \rangle$ by $\langle H \cup \{pl_1\}, pl_1 \rangle$ and $\langle H \cup \{pl_2\}, pl_2 \rangle$, because $H \cup \{pl_1, pl_2\} \cup \{pl_3\} \vdash_\tau \bot$ and $pl_3 \neq pl_4$.
- One-to-one rebutal: In Figure 1(b), $\langle H \cup \{\mathbf{g}(q_3)\}, \mathbf{g}(q_3) \rangle$ attacks $\langle H \cup \{\mathbf{q}(p_3)\}, \mathbf{q}(p_3) \rangle$ and the attack is a rebutal because $\mathbf{k}(\mathbf{g}(q_3) \wedge \mathbf{q}(p_3) \rightarrow \bot) = \mathbf{k}(\psi_1)$ is in the r-net shown in that figure, i.e., $\mathbf{k}(\psi_1) \in H$.
- One-to-one undercut: In Figure 1(b), $\langle H \cup \{\mathbf{g}(q_3)\}, \mathbf{g}(q_3) \rangle$ undercuts $\langle H \cup \{\mathbf{g}(q_1), \mathbf{g}(q_2), \mathbf{g}(q_3)\}, \mathbf{g}(q) \rangle$, because $\mathbf{k}(\mathbf{g}(q_3) \wedge \mathbf{q}(p_3) \rightarrow \bot) = \mathbf{k}(\psi_1)$ is in the r-net shown in that figure, i.e., $\mathbf{k}(\psi_1) \in H$ and $\mathbf{g}(q_3) \neq \mathbf{g}(q)$. ■

We said above that $\xrightarrow{A}$ is not primitive, and Proposition III.39 justifies that claim by showing that two consistent sets of potential solutions will attack each other if and only if parts of the potential solutions in the two sets

have microsolutions such that the union of their bases is inconsistent. Stated otherwise, attack between some internally consistent structures (be they microsolutions, or sets of microsolutions, i.e., potential solutions, or sets of potential solutions) simply reflects the presence of conflict between S-nodes in these structures.

Proposition III.39. *There is an attack relation between two potential solutions $\mathcal{S}_A$ and $\mathcal{S}_B$ of $\bar{\mathcal{R}}$, such that $\bar{S}_A \in Base(\mathcal{S}_A)$ and $\bar{S}_B \in Base(\mathcal{S}_B)$ if and only if there is a conflict between $\bar{S}_A$ and $\bar{S}_B$ in an $\bar{\mathcal{R}}$.*

Proof: (Trivial.) There is a conflict between $\bar{S}_A$ and $\bar{S}_B$ in an $\bar{\mathcal{R}}$ iff $H \cup \bar{S}_A \cup \bar{S}_B \vdash_\tau \bot$. Since H is in every potential solution of $\bar{\mathcal{R}}$, it follows that if $\bar{S}_A \in Base(\mathcal{S}_A)$ and $\bar{S}_B \in Base(\mathcal{S}_B)$, then $Base(\mathcal{S}_A) \cup Base(\mathcal{S}_B) \vdash_\tau \bot$, i.e., $\mathcal{S}_A$ and $\mathcal{S}_B$ attack each other. ■

Proposition III.39 concerns only cases in which the two sets of potential solutions that attack each other are singletons. This case is chosen to keep notation simple, and because it is obvious how to extend Proposition III.39 to the case in which the two sets of potential solutions that attack each other are not singletons.

2) Techne Argumentation Framework: Having defined the attack relation between consistent sets of potential solutions, we now turn to the fundamental role of argumentation in the definition and identification of robust potential solutions. This role becomes apparent by defining what we will call *Techne argumentation framework*.

Definition III.40. ***Techne Argumentation Framework.*** *$(\mathsf{Arg}, \xrightarrow{A})$ is a Techne argumentation framework, where:*

- *$\mathsf{Arg} = \{\mathsf{A}_1, \ldots, \mathsf{A}_n\}$;*
- *$\forall 1 \leq k \leq n$, $\mathsf{A}_k = \{\mathcal{S}_1, \ldots, \mathcal{S}_n \mid \bigcup_{i=1}^{n} Base(\mathcal{S}_i) \nvdash_\tau \bot\}$;*
- *every $\mathcal{S}_i$ in every A_k is a potential solution;*
- *$\xrightarrow{A} \subseteq \mathsf{Arg} \times \mathsf{Arg}$.*

Remark III.41. As a convention, we call Arg the *set of arguments*, and every member A_k of that set is called an *argument*. Every argument is a consistent set of potential solutions. $\xrightarrow{A}$ is the attack relation between arguments. Remark that every structure made up of microsolutions that are together not inconsistent is an argument: an argument can be a single microsolution, a conflict-free set of microsolutions (i.e., a potential solution), or a consistent set of potential solutions. ■

A $\mathsf{TAF} = (\mathsf{Arg}, \xrightarrow{A})$ corresponds to an abstract argumentation framework $AF(\mathsf{TAF}) = (AR, attacks)$ as follows:

1) $AR = \mathsf{Arg}$, and
2) $\xrightarrow{A} \equiv attacks$.

To be sure that this correspondence is meaningful, we must show that there is equivalence between members of the semantic domain of the the Techne argumentation framwork and the corresponding abstract argumentation framework. We will show this later (§III-D4) in relation to Dung's abstract argumentation.

What remains to be done here in relation to TAFs is to substantiate the claim made earlier, that any $\bar{\mathcal{R}}$ can be redefined as an abstract argumentation framework. Obviously, such redefinition is only interesting as long as nothing gets lost in the process.

Recall that $\bar{\mathcal{R}}$ is a set of S-nodes and R-nodes, and that it lacks all preference, is-optional, and is-mandatory relations. The R-nodes in $\bar{\mathcal{R}}$ are consequently either `I`-nodes that capture steps in proofs (i.e., inference steps), and `C`-nodes which indicate which sets of S-nodes are inconsistent. All S-nodes $\bar{S}$ of $\bar{\mathcal{R}}$ are partitioned onto source and derived S-nodes: $Source(\bar{\mathcal{R}})$ are all S-nodes which, in terms of graph syntax, have no incoming edges in $\bar{\mathcal{R}}$, or in terms of symbolic syntax, they are asserted nodes, those for which no proofs are given. An S-node *pl* is derived iff $Source(\bar{\mathcal{R}}) \vdash_\tau pl$. Since we do not prove Horn domain assumptions, $H \subseteq Source(\bar{\mathcal{R}})$, and hence, $\bar{S} \setminus Source(\bar{\mathcal{R}})$ is the set of derived S-nodes.

The definition of the microsolution concept (cf., Definition III.20) tells us that in $\langle \bar{S}_i, pl_i \rangle$, $\bar{S}_i \subseteq Source(\bar{\mathcal{R}})$. It follows that every microsolution of $\bar{\mathcal{R}}$ is a proof from some subset of the source S-nodes of $\bar{\mathcal{R}}$. We know that every potential solution of $\bar{\mathcal{R}}$ is made up of microsolutions of $\bar{\mathcal{R}}$ (cf., Definition III.23), so that every potential solution is made up of proofs from $Source(\bar{\mathcal{R}})$. It is consequently clear that once we know $Source(\bar{\mathcal{R}})$, we can build the Techne argumentation framework for that $\bar{\mathcal{R}}$ — no other information is needed. We synthesize this in Proposition III.42.

Proposition III.42. *(1) There is for every $\bar{\mathcal{R}}$ a unique corresponding Techne argumentation framework $\mathsf{TAF}(\bar{\mathcal{R}})$, and (2) for every Techne argumentation framework, there is a unique corresponding $\bar{\mathcal{R}}$.*

Proof: (Trivial, from definitions.)

1) Given an $\bar{\mathcal{R}}$, we know $Source(\bar{\mathcal{R}})$. Let $M = \{\langle \bar{S}_i, pl_i \rangle \mid \bar{S}_i \subseteq Source(\bar{\mathcal{R}})\}$, i.e., M is the set of all microsolutions in $\bar{\mathcal{R}}$. Obviously, every potential solution $\mathcal{S}_j$ of $\bar{\mathcal{R}}$ is a subset of or equal to M, i.e., $\forall j$, $\mathcal{S}_j \subseteq M$. It follows that $\mathsf{Arg} \subseteq \wp M$, and hence there is for every $\bar{\mathcal{R}}$ a unique corresponding Techne argumentation framework $\mathsf{TAF}(\bar{\mathcal{R}}) = (\mathsf{Arg}, \xrightarrow{A})$.
2) Given a Techne argumentation framework $\mathsf{TAF}(\bar{\mathcal{R}}) = (\mathsf{Arg}, \xrightarrow{A})$, we know that every member of Arg is a set of potential solutions, and thus, a set of microsolutions. Let $\{\langle \bar{S}_1, pl_1 \rangle, \ldots, \langle \bar{S}_n, pl_n \rangle\}$ be the set of microsolutions appearing in potential solutions, and thus in Arg. It follows that we can define an $\bar{\mathcal{R}}$ by defining $Source(\bar{\mathcal{R}}) = \bigcup_{i=1}^{n} \bar{S}_i$. It follows that for every Techne argumentation framework, there is a unique corresponding $\bar{\mathcal{R}}$.

■

3) Robust Potential Solutions: The semantic domain of an $\bar{\mathcal{R}}$ is made up of potential solutions, themselves sets of consistent microsolutions. Some of the potential solutions are subsets of others. Stated otherwise, the semantic domain of an $\bar{\mathcal{R}}$ is the structre $(\mathsf{PS}(\bar{\mathcal{R}}), \subseteq)$, where $\mathsf{PS}(\bar{\mathcal{R}})$ is the set of all potential solutions of $\bar{\mathcal{R}}$, i.e.:

$$\mathsf{PS}(\bar{\mathcal{R}}) = \{\mathcal{S}_i \mid 1 \leq i \leq n, \\ \forall i, j,\ \langle \bar{S}_j, pl_j \rangle \in \mathcal{S}_i,\ \bar{S}_j \subseteq \mathit{Source}(\bar{\mathcal{R}})\}$$

and $\subseteq$ is a binary relation over $\mathsf{PS}(\bar{\mathcal{R}})$ that is reflexive, antisymmetric, and transitive (i.e., a partial order). We used (cf., Definition III.25) to define a preferred potential solution as any potential solution that is not a strict subset of another in $\mathsf{PS}(\bar{\mathcal{R}})$, i.e., $\mathcal{S}_i$ is a preferred potential solution iff $\nexists \mathcal{S}_j \in \mathsf{PS}(\bar{\mathcal{R}})$ such that $\mathcal{S}_i \subset \mathcal{S}_j$.

Neither the potential solution nor the preferred potential solution concepts have much to do with robustness. This is because *the only requirement that a set of microsolutions must satisfy in order to be called a potential solution is that it is not inconsistent*. In Examples III.33–III.34 we had (preferred) potential solutions which were not robust: e.g., $\mathcal{S}_p$ in Example III.33 and $\mathcal{S}$ in Example III.34.

To introduce the notion of robustness, we define a robust potential solution as follows.

Definition III.43. ***Robust Potential Solution.*** *A set of arguments is a robust potential solution, $\mathfrak{S}$ iff $\mathfrak{S}$ defends every one of its members.*

Definition III.44. ***Defense.*** *Given $\mathsf{TAF}(\bar{\mathcal{R}}) = (\mathsf{Arg}, \xrightarrow{A})$, a set of arguments $\mathfrak{S} \subseteq \mathsf{Arg}$ defends $\mathsf{A} \in \mathsf{Arg}$ iff $\forall \mathsf{A}' \in \mathsf{Arg}$ such that $\mathsf{A}' \xrightarrow{A} \mathsf{A}$, then $\exists \mathsf{A}'' \in \mathfrak{S}$ and $\mathsf{A}'' \xrightarrow{A} \mathsf{A}'$.*

Example III.45. In Example III.33, let $\mathsf{A} = \{\mathcal{S}_p\}$, $\mathsf{A}' = \{\mathcal{S}_q\}$ and $\mathsf{A}'' = \{\mathcal{S}_r\}$, so that $\mathsf{A}' \xrightarrow{A} \mathsf{A}$ and also $\mathsf{A}'' \xrightarrow{A} \mathsf{A}'$. It follows that $\mathfrak{S} = \{\mathcal{S}_{pr}\}$ is a robust potential solution, because it defends its members from attacking arguments.

Proposition III.46. *Every robust potential solution $\mathfrak{S}$ of $\bar{\mathcal{R}}$ is a potential solution, i.e., $\mathfrak{S} \in \mathit{PS}(\bar{\mathcal{R}})$.*

Proof: (Obvious.) A robust potential solution $\mathfrak{S}$ is a set of arguments that defends every one of its members, so that no argument in $\mathfrak{S}$ attacks another argument in $\mathfrak{S}$. $\mathfrak{S}$ is thus a set of arguments that is not inconsistent, and hence a set of potential solutions that are not inconsistent. A set of potential solutions that is not inconsistent is itself a potential solution. ∎

Proposition III.46 states that some potential solutions are in fact robust potential solutions, i.e., that robust potential solutions are a subset of all potential solutions, as a robust potential solution is not inconsistent (hence, every robust potential solution is a potential solution) and it satisfies an additional property, namely, that it defends its members from attacking arguments. It follows that some $\mathfrak{S}$ is a preferred robust potential solution iff there is in $\mathsf{PS}(\bar{\mathcal{R}})$ no other robust potential solution $\mathfrak{S}'$ such that $\mathfrak{S} \subset \mathfrak{S}'$.

Once we have defined robust potential solutions, we are evidently interested in building them (e.g., from potential solutions), and in moving from only a robust potential solution to a preferred robust potential solution. Both of these tasks proceed in a way that is analogous to that applied to move from a potential solution to a preferred potential solution. We saw earlier (cf., §III-C) we can add a microsolution to a potential solution $\mathcal{S}$ iff $\mathcal{S}$ together with the microsolution are not inconsistent. Rather than making A being acceptable w.r.t. a $\mathfrak{S}$ if $\mathfrak{S} \cup \{\mathsf{A}\}$ is not inconsistent, we say that A is accetable w.r.t. $\mathfrak{S}$ iff $\mathfrak{S}$ defends A from the arguments that attack A.

Definition III.47. ***Acceptable Argument.*** *An argument A is acceptable w.r.t. a robust potential solution $\mathfrak{S}$ iff $\mathfrak{S}$ defends A from every argument that attacks A.*

Following Definition III.47, if we are given a robust potential solution $\mathfrak{S}$, we can verify whether it it is also a preferred robust potential solution by checking if there are arguments outside $\mathfrak{S}$ that are acceptable w.r.t. $\mathfrak{S}$. If there are no such arguments, then $\mathfrak{S}$ is a preferred robust potential solution; otherwise, we can add these arguments to $\mathfrak{S}$ to obtain a preferred potential solution.

4) Fundamental Role of Abstract Argumentation in Techne: We introduced up to this point a semantic domain for attitude-free r-nets, and defined a number of solution concepts, the instances of which are members of the semantic domain. Potential solution is the basic one, the specializations of which are the preferred potential solution and robust potential solution concepts. We finally gave conditions for an instance of the robust potential solution concept to be also an instance of the concept of preferred robust potential solution. Because finding robust potential solutions requires us to check whether a potential solution defends its members from outside attacks, we say that semantics are dialectical.

We said earlier that a Techne argumentation framework $\mathsf{TAF}(\bar{\mathcal{R}})$ for an $\bar{\mathcal{R}}$ corresponds to an abstract argumentation framework $AF(\mathsf{TAF}(\bar{\mathcal{R}}))$. The conversion from the former to the latter is meaningful if and only if members of their semantic domains are equivalent. To show that the conversion is meaningful is to show that an TAF is a special case of Dung's abstract argumentation framework AF, and thus, intuitively, that the search for robust potential solutions is a dialectical process, in which arguments are confronted to determine which of them should be accepted, and which rejected.

Dung's abstract argumentation framework [9] is a structure of the form $AF = (AR, attacks)$, where AR is a set of primitive (undefined) structures called *arguments*, and *attacks* is a binary relation between arguments. The semantic domain of AF is made up of extensions, which are conflict-free sets of arguments, i.e., sets of arguments which do not attack each other. The central notion is that of acceptability of an argument, whereby $A \in AR$ is acceptable w.r.t. a set X of arguments iff for each argument $B \in AR$: if B attacks A then B is attacked by S. A conflict-free set X of arguments is admissible iff it each argument in X is acceptable w.r.t. X. It is enough to know this in order to offer the following proposition.

Proposition III.48. *There is for every Techne argumentation framework $\mathsf{TAF}(\bar{\mathcal{R}})$ a corresponding Dung's abstract argumentation framework $AF(\mathsf{TAF}(\bar{\mathcal{R}}))$ such that every robust potential solution of $\bar{\mathcal{R}}$ is an admissible extension of $AF(\mathsf{TAF}(\bar{\mathcal{R}}))$.*

Proof: (Trivial, from definitions.) A robust potential solution of $\bar{\mathcal{R}}$ is a set of arguments that defends every one of its members (cf., Definition III.43). An admissible extension of an abstract argumentation framework is a conflict-free set of arguments in which every argument is acceptable w.r.t. to that set. As a robust potential solution defends all of its arguments, it is a conflict-free set of arguments and every argument in it is acceptable w.r.t. to that robust potential solution. There is consequently for every robust potential solution of $\bar{\mathcal{R}}$ an admissible extension of $AF(\mathsf{TAF}(\bar{\mathcal{R}}))$. ■

IV. Related Work

Surveys of RE research — from van Lamsweerde [10] and Robinson et al. [11] in particular — confirm Zave and Jackson's [7] prior observation that the field had already in the 1980s left behind simplistic approaches to understanding what a system-to-be would do in favor of novel and varied terminology, methods, languages, tools, and issues considered to be critical. One constant is the observation that RMLs play a central role in both research and practice of RE. It does not require much knowledge of the field to see that many research efforts that fall within Zave's classification [12] relate in one way or another to one or more RMLs; e.g., elicitation of information from stakeholders, validation, specification, checks for incompleteness and inconsistency, all suppose that some model of requirements is available.

Despite the important position that RMLs play in RE, there are no widely-accepted and precise standards that a formalism must satisfy in order to be called an RML. The evolution of RMLs seems to be one of testing of and converging on similar ideas (e.g., we find refinement in one way or another in most, if not all RMLs), rather then the design of formalisms following clear desiderata. We will highlight some of these key ideas in the rest of this section and position Techne in relation to them.

A. Specification Languages as RMLs

Highly developed languages for the specification of the properties of a system-to-be — i.e., formal methods such as Z, VDM, Larch, temporal logic, CSP, transition axioms, among others (e.g., [13], [14]) — have been available alongside most RMLs, and they have been used to perform some of the tasks of RMLs. However, it was recognized and now seems to be common knowledge, that there is a "need to distinguish between *specification* languages, [...] and *modeling* languages, which aspire to offer facilities for the description of settings, or more precisely, humans' knowledge/beliefs about these worlds [i.e., environment of the system-to-be]. This is a philosophical and psychological point which has profound implications for requirements language designers and users alike, as well as the requirements discourse." [15] If one sees — as in this paper — an RML as made of three parts, then specification languages seem inconvenient despite their sophistication, for they fail to include principles for the organization of their artifacts in a way that facilitates the definition of the requirements problem, its understanding by, clarification for, and negotiation between the systems' stakeholders. That organization is an aim of the ontology which in an RML states what kinds of information the engineer is dealing with. The ontology serves as a filter through which to distinguish the roles that pragmatically different kinds of information have with regards to the properties of the environment and of the system-to-be. The second part of the RML, the models that RMLs produce facilitate the communication between the stakeholders and engineers. Finally, the reasoning that can be performed over these models answers relevant questions on the presence of inconsistencies and of the solutions to the requirements problem.

Though requirements can ultimately be rewritten as states and transitions via specification languages when resources allow, this changes in no way the prima facie evidence that stakeholders do not provide information in such a form that system's states and transitions can immediately be recognized. RMLs have the difficult task to bridge the messy informal first steps in information systems engineering and those in which rigorous application of formal methods, and thus specification languages becomes feasible.

B. Original RML

That there is more to writing requirements than functional specification was recognized in the original RML [4] (hereafter ORML), "a notation for requirements modeling which combines object-orientation and organization, with an assertional sublanguage used to specify constraints and deductive rules" [15]. Formal semantics is given to ORML via a mapping from its descriptions to assertions in first order logic (hereafter FOL). One thereby obtains facilities for the structuring and organization of FOL theories. The ontology in ORML distinguishes between entities, activities, and assertions. The ontology was judged limited [15] and responses to limitations went in two directions. RMLs such as KAOS and *i** took the direction in which the ontology remains fixed (i.e., one cannot add or remove concepts when applying the RML) but include more concepts, designed to cover concerns such as the desires of the system's stakeholders (see below). The other direction was adopted in Telos [16] and consists of leaving the ontology undefined, while having in the language the facilities needed to define the ontology. The second approach is more expressive, but its abstraction makes it difficult to provide methodological guidance which can be given when a fixed set of concepts is known and manipulated every time the language is used.

C. KAOS

"The overall approach taken in KAOS has three components: (i) a conceptual model for acquiring and structuring requirements models, with an associated acquisition language, (ii) a set of strategies for elaborating requirements models in this framework, and (iii) an automated assistant to provide guidance in the acquisition process according to such strategies." [5] The conceptual model specifies the ontology in KAOS, which includes a number of concepts (object, operation, agent, goal, obstacle, requisite/requirement/assumption, scenario) and relations (specialization, refinement, conflict, operationalization, concern, and so on) [5], [17], [18]. A KAOS model of requirements instantiates the concepts, relates these instances, declares instances' properties which are relevant to the elaboration/transformation of the model, and allows the engineer to formally define the instance as a theory of linear temporal FOL. In light of ORML and specification languages, KAOS can be understood as a framework (i.e., a combination of an RML and of a methodology for the use of that RML) which is defined on top of linear temporal FOL that serves as a specification language in KAOS. ORML and KAOS are thus similar, as both aim to facilitate the organization of logical theories via the classification of theories as instances of predefined concepts, and the definition of concept instances as logical theories.

D. I-Star (i*), Tropos, and Formal Tropos

*i** [19], [6] is an RML that distinguishes itself strongly from those mentioned above both in its design and its focus. In terms of design, *i** is not defined on top of a specification language. The focus of *i** is on the interdependencies of actors within a socio-technical system, their individual and joint goals, tasks, and available or necessary resources, the roles they occupy. A model of requirements made with *i** aims to be a snapshot of the intentional states of actors, along with what roles they adopt, and how they depend on each other for the satisfaction of individual and joint goals, the performance of tasks, and use of resources. The system-to-be or its components are actors alongside individuals and groups. In contrast to both ORML and KAOS, the engineer cannot formally verify the satisfaction of requirements (i.e., check if a system's properties satisfy goals [10]) via an *i** requirements model; the closest the engineer can do is validate them instead via informal discussion with the stakeholders. It is perhaps this departure from specification languages as foundations for RMLs that led to considerable work on *i**. It is a lightweight RML, the non-formal character of which makes it easy to learn and use, a critical feature given that requirements must be validated by stakeholders who cannot be expected to manipulate artifacts produced with specification languages.

Tropos [20], a methodology for information systems engineering uses *i** as its RML at the very first steps of the RE process, when it is impractical to start writing formal theories in a variant of FOL or another formalism. Once *i** models of the system-to-be within its organizational environment are available, Tropos explains how to proceed towards data and behavior models of the system-to-be. Formal Tropos [21] continued the tradition of giving formal semantics to RMLs by mapping instances of *i** concepts and relations between them (i.e., *i** requirements models) to theories of linear temporal FOL. An important contribution of Formal Tropos was to "demonstrate that formal analysis techniques are useful during early development phases. The novelty of the approach lies in extending model checking techniques — which rely mostly on design-inspired specification languages — so that they can be used for early requirements modeling and analysis." [21]

E. Techne

Techne is an RML designed for use in the very early steps of the RE process, when the requirements problem for the system-to-be is still unclear. The purpose of Techne is to help the structuring of the requirements problem and preliminary identification of alternative solutions along with their comparative evaluation. This early use makes it necessary to stay with simple means of knowledge representation, and straightforward rules of inference, which led us to adopt propositional logic instead of a predicate logic, and to adopt resolution as the only rule of inference.

Techne is consequently quite different from ORML, KAOS, and Formal Tropos. ORML obtains formal semantics via the mapping of its models/descriptions to FOL. In KAOS requirements models, formal definitions of concept instances have formal semantics via their writing in linear temporal FOL, and some relations, such as refinement do have formal definitions. In Formal Tropos, instances of *i** concepts are — similarly to KAOS — defined in linear temporal FOL. ORML and KAOS are object-oriented, featuring the specialization relation. Techne is not object-oriented and does not incorporate the specialization relation. Atoms in Techne are propositions, and given the purpose of Techne, these propositions are likely to be written as sentences of natural language.

Techne and *i** differ in several respects. *i** has no notion of conflict, preference or mandatory/optional requirements, no formal semantics, and thus has no precise notion of what a solution to the requirements problem is. Alternative decompositions of a goal are compared in terms of their contributions to softgoals. Techne keeps softgoals, but due to the vagueness of softgoal instances [22], [1] we require that they are approximated, i.e., "refined" by other non-softgoals, among which preference relations can be added to indicate which satisfy the softgoal in more desirable ways than others. Techne has no concepts pertaining to actors and roles.

Giorgini et al. [23] recognized the need to formalize goal models so as to automatically evaluate the (degree of) satisfaction of goals. Their goal models are AND/OR

graphs, in which nodes are goals, and a number of relations is provided to indicate if the interaction is positive or negative (i.e., how the satisaction of a goal influences the satisfaction of the other goal related to it), as well as to specify the strength of the interaction. Techne uses preferences to indicate in the relative degrees of satisfaction (cf., Example II.5), while quantiative estimates of satisfaction levels cannot be used. Goal models from Giorgini et al. do not incorporate the notion of conflict as inconsistency, they do not include concepts other than goals, cannot distinguish optional from mandatory requirements, and have no notion of robust solutions.

V. Conclusions

Techne is an RML designed to assist knowledge representation and decision making during the very early stages of RE, when the requirements problem is being structured and its alternative potential solutions explored. The language has formal semantics, with paraconsistent and nonmonotonic reasoning. Requirements problem and its solutions obtain mathematically formal definitions in Techne. A variety of concepts can be instantiated to capture requirements in Techne r-nets.

Techne does not integrate specialized facilities for knowledge representation, such as specialization, nor does it feature social concepts, such as actors or roles. Techne complements existing RMLs which do include such notions. Techne assists the decision-making towards a solution to the requirements problem, which, once available as a set of requirements can be subjected to analyses that *i** supports, such as the distribution of responsibility and the analysis of dependencies in the realization of that solution. Subsequent steps may involve KAOS, for the definition of requirements concepts as theories of linear temporal FOL, allowing thus assisted detailed refinement and verification towards the operationalization of the solution via data and behavior models appropriate at later steps of information systems engineering.

Ongoing work on Techne focuses on the definition and testing of efficient reasoning methods for the search of solutions in r-nets, as well as on tool support for modeling and reasoning.

References

[1] I. J. Jureta, J. Mylopoulos, and S. Faulkner, "Revisiting the core ontology and problem in requirements engineering," in *16th IEEE Int. Requirements Engineering Conf.*, 2008.

[2] S. J. Greenspan, J. Mylopoulos, and A. Borgida, "Capturing more world knowledge in the requirements specification," in *ICSE*, 1982.

[3] D. A. Marca and C. L. McGowan, *SADT: structured analysis and design technique*. McGraw-Hill, Inc., 1987.

[4] S. J. Greenspan, A. Borgida, and J. Mylopoulos, "A requirements modeling language and its logic," *Inf. Syst.*, vol. 11, no. 1, pp. 9–23, 1986.

[5] A. Dardenne, A. van Lamsweerde, and S. Fickas, "Goal-directed requirements acquisition," *Sci. Comput. Program.*, vol. 20, no. 1-2, pp. 3–50, 1993.

[6] E. Yu, "Towards modeling and reasoning support for early requirements engineering," in *Proceedings of the IEEE International Symposium on Requirements Engineering*, 1997.

[7] P. Zave and M. Jackson, "Four dark corners of requirements engineering," *ACM T. Softw. Eng. Methodol.*, vol. 6, no. 1, pp. 1–30, 1997.

[8] R. Darimont and A. van Lamsweerde, "Formal refinement patterns for goal-driven requirements elaboration," in *SIGSOFT FSE*, 1996.

[9] P. M. Dung, "On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and n-person games," *Artif. Intell.*, vol. 77, no. 2, pp. 321–358, 1995.

[10] A. van Lamsweerde, "Goal-oriented requirements engineering: A guided tour," in *5th IEEE International Symposium on Requirements Engineering (RE)*, 2001, p. 249.

[11] W. N. Robinson, S. D. Pawlowski, and V. Volkov, "Requirements interaction management," *ACM Comput. Surv.*, vol. 35, no. 2, pp. 132–190, 2003.

[12] P. Zave, "Classification of research efforts in requirements engineering," *ACM Comput. Surv.*, vol. 29, no. 4, pp. 315–321, 1997.

[13] J. M. Wing, "A specifier's introduction to formal methods," *IEEE Computer*, vol. 23, no. 9, pp. 8–24, 1990.

[14] E. M. Clarke and J. M. Wing, "Formal methods: state of the art and future directions," *ACM Comput. Surv.*, vol. 28, no. 4, pp. 626–643, 1996.

[15] S. Greenspan, J. Mylopoulos, and A. Borgida, "On formal requirements modeling languages: Rml revisited," in *ICSE '94: Proceedings of the 16th international conference on Software engineering*. Los Alamitos, CA, USA: IEEE Computer Society Press, 1994, pp. 135–147.

[16] J. Mylopoulos, A. Borgida, M. Jarke, and M. Koubarakis, "Telos: representing knowledge about information systems," *ACM Trans. Inf. Syst.*, vol. 8, no. 4, pp. 325–362, 1990.

[17] A. van Lamsweerde, R. Darimont, and E. Letier, "Managing conflicts in goal-driven requirements engineering," *IEEE Trans. Software Eng.*, vol. 24, no. 11, pp. 908–926, 1998.

[18] A. van Lamsweerde and E. Letier, "Handling obstacles in goal-oriented requirements engineering," *IEEE Trans. Software Eng.*, vol. 26, no. 10, pp. 978–1005, 2000.

[19] E. S. K. Yu and J. Mylopoulos, "Understanding "why" in software process modelling, analysis, and design," in *ICSE*, 1994, pp. 159–168.

[20] J. Castro, M. Kolp, and J. Mylopoulos, "Towards requirements-driven information systems engineering: the tropos project," *Inf. Syst.*, vol. 27, no. 6, pp. 365–389, 2002.

[21] A. Fuxman, L. Liu, J. Mylopoulos, M. Roveri, and P. Traverso, "Specifying and analyzing early requirements in tropos," *Requir. Eng.*, vol. 9, no. 2, pp. 132–150, 2004.

[22] I. J. Jureta, S. Faulkner, and P.-Y. Schobbens, "A more expressive softgoal conceptualization for quality requirements analysis," in *Proceedings of the 25th International Conference on Conceptual Modelling (ER'06)*, 2006.

[23] P. Giorgini, J. Mylopoulos, E. Nicchiarelli, and R. Sebastiani, "Formal reasoning techniques for goal models," *J. Data Semantics*, vol. 1, pp. 1–20, 2003.