Proceedings

2010 IEEE Seventh International Conference on Services Computing

*Miami, Florida*
5–10 July 2010

**Sponsored by**
IEEE Computer Society Technical Committee on Services Computing (TC-SVC)

**Supported by**
IBM Research
SAP

Los Alamitos, California
Washington • Tokyo
Table of Contents

Message from the General and Program Chairs .......................................................... xiii
Organizing Committee .................................................................................................. xiv
Technical Program Committee .................................................................................. xvi
IEEE Computer Society Technical Committee on Services Computing ................. xx

Research Session 1—Choreography of Business Process Management

A Model for Visual Specification of E-contracts .......................................................... 1

Enrique Martínez, Gregorio Díaz, M. Emilia Cambronero, and Gerardo Schneider

Preserving Data Flow Correctness in Process Adaptation ......................................... 9

Wei Song, Xiaoxing Ma, S.C. Cheung, Hao Hu, and Jian Lü

Process Trace Identification from Unstructured Execution Logs ............................... 17

Nirmi Desai, Anuradha Bhamidipaty, Bhuvan Sharma, Virendra K. Varshneya,
Mitesh Vasa, and Shailabh Nagar

Research Session 2—Modeling and Abstraction of Services Process

Dynamic Service Invocation Control in Service Composition Environments .......... 25

Donghui Lin, Yohei Murakami, and Masahiro Tanaka

BPEL Remote Objects: Integrating BPEL Processes into Object-Oriented
Applications .................................................................................................................. 33

Marvin Ferber, Thomas Rauber, and Sascha Hunold

Co-Taverna: A Tool Supporting Collaborative Scientific Workflows ....................... 41

Jia Zhang
Research Session 3—Business Oriented and Enabled SOA

Business Driven Consolidation of SOA Implementations .................................................................49
   Biplav Srivastava and Pietro Mazzoleni
Detection of Semantically Equivalent Fragments for Business Process Model Change Management ...............................................................................................................57
   Christian Gerth, Markus Luckey, Jochen M. Küster, and Gregor Engels
A Novel Approach to Improving the Efficiency of Storing and Accessing Small Files on Hadoop: A Case Study by PowerPoint Files .................................................................65
   Bo Dong, Jie Qiu, Qinghua Zheng, Xiao Zhong, Jingwei Li, and Ying Li

Research Session 4—Cloud Services

A Formal Service Contract Model for Accountable SaaS and Cloud Services ....................................73
   Joe Zou, Yan Wang, and Kwei-Jay Lin
Accountability as a Service for the Cloud .............................................................................................81
   Jinhui Yao, Shipping Chen, Chen Wang, David Levy, and John Zic
Graph-Based Cloud Service Placement ..................................................................................................89
   Anca Sailer, Michael R. Head, Andrzej Kochut, and Hidayatullah Shaikh

Research Session 5—QoS and Services Composition

Metaheuristic Optimization of Large-Scale QoS-aware Service Compositions ......................................97
   Florian Rosenberg, Max Benjamin Müller, Philipp Leitner, Anton Michlmayr,
   Athman Bouguettaya, and Schahram Dustdar
Xi-Calculus: A Calculus for Service Interactions .................................................................................105
   Vineet Rajani, Arun Kumar, and Dharanipragada Janakiram
A Service-Oriented Framework for GNU Octave-Based Performance Prediction ..................................114
   George Kousiouris, Dimosthenis Kyriazis, Kleopatra Konstanteli, Spyridon Gogouvitis,
   Gregory Katsaros, and Theodora Varvarigou

Research Session 6—QoS Selection and Matchmaking

QoS Analysis and Service Selection for Composite Services ....................................................................122
   Huiyuan Zheng, Jian Yang, and Weiliang Zhao
A Heuristic Algorithm for Trust-Oriented Service Provider Selection in Complex Social Networks ..........................................................130
   Guanfeng Liu, Yan Wang, Mehmet A. Orgun, and Ee-Peng Lim
An Information Model for the Provisioning of Network Connections Enabling Customer-Specific End-to-End QoS Guarantees ..................................................................................138
   Mark Yampolskiy, Wolfgang Hommel, Patricia Marcu, and Matthias K. Hamm
Research Session 7—Design and Modeling of SOA

SecureSOA—Modelling Security Requirements for Service-Oriented Architectures ................................................................. 146

   Michael Menzel and Christoph Meinel

Service Supervision for Service-Oriented Collective Intelligence ........................................................................................................ 154

   Masahiro Tanaka, Yohei Murakami, Donghui Lin, and Toru Ishida

Research Session 8—Semantic Web Services

Meta-Modeling of Semantic Web Services .............................................................................................................................. 162

   Roberto De Virgilio

Improve Service Interface Adaptation Using Sub-ontology Extraction ............................................................................................. 170

   Lu Jin, Jian Wu, Jianwei Yin, Yin Li, and Shuiguang Deng

Distributed Storage and Querying Techniques for a Semantic Web of Scientific Workflow Provenance .......................................................... 178

   John Abraham, Pearl Brazier, Artem Chebotko, Jaime Navarro, and Anthony Piazza

Research Session 9—Service Discovery and Delivery

Discovering Shared Services from Cross-Organizational Software Specifications ......................................................................................... 186

   Alexander Yale-Loehr, Ian D. Schlesinger, Aubrey J. Rembert, and M. Brian Blake

Analyzing Communities vs. Single Agent-Based Web Services: Trust Perspectives .................................................................................. 194

   Babak Khosravifar, Jamal Bentahar, Ahmad Moazin, Zakaria Maamar, and Philippe Thiran

A Service Packaging Platform for Delivering Services ...................................................................................................................... 202

   Muhammad Adeel Talib, Alan Colman, Jun Han, Justin King, and Malinda Kapuruge

Research Session 10—Service Composition, Coordination, and Management

A Value-Driven Approach for the Determination of Global Optimization Objective of Service Composition ........................................................................... 210

   Zhongjie Wang, Xiaofei Xu, Dianhui Chu, and Chao Ma

Applying Bargaining Game Theory to Web Services Negotiation ........................................................................................................ 218

   Xianrong Zheng, Patrick Martin, Wendy Powley, and Kathryn Brohman

A Continuous Long Running Batch Orchestration Model for Workflow Instance Migration .................................................................................. 226

   David Frank, Liана Fong, and Linh Lam
# Application and Industry Session 1—Services Composition Methodology

Describing and Verifying Web Service Composition Using TLA Reasoning ................................................. 234
*Hongbing Wang, Qianzhao Zhou, and Yanqi Shi*

Dynamic Composition of Services in Sensor Networks ............................................................................ 242
*Sahin Cem Geyik, Boleslaw K. Szymanski, Petros Zerfos, and Dinesh Verma*

Discovering Business Process Model from Unstructured Activity Logs .................................................. 250
*Rahul Kumar, Chiranjib Bhattacharyya, and Virendra Varshneya*

# Application and Industry Session 2—Services Selection and Matchmaking

Utilizing the Waiting-Time Criterion for Selecting Services in a Composition ........................................ 258
*Abhishek Srivastava and Paul G. Sorenson*

ViPen: A Model Supporting Knowledge Provenance for Exploratory Service Composition ..................... 265
*Weilong Ding, Jing Wang, and Yanbo Han*

A QoS Query Language for User-Centric Web Service Selection ............................................................. 273
*Delnavaz Mobedpour, Chen Ding, and Chi-Hung Chi*

# Application and Industry Session 3—Policy Driven Business Process Generation

Analyst-Mediated Contextualization of Regulatory Policies ........................................................................ 281
*George Koliadis, Nirmi V. Desai, Nanjungud C. Narendra, and Aditya K. Ghose*

Measuring Compliance and Deviations in a Template-Based Service Contract ........................................ 289
*Vijil Chenthamarakshan, Rafah A. Hosn, Shajith Ikbal, Nanda Kambhatla, Debpriyo Majumdar, and Soumitra Sarkar*

Towards a Novel Web Services Standard-Supported CDN-P2P Loosely-Coupled Hybrid and Management Model ........................................................... 297
*ZhiHui Lu, Jie Wu, and WeiMing Fu*

# Application and Industry Session 4—Trust Management

Universal Identity Management Model Based on Anonymous Credentials .............................................. 305
*Yang Zhang and Jun-Liang Chen*

A Trust Ontology for Semantic Services .................................................................................................. 313
*Wanita Sherchan, Surya Nepal, Jonathon Hunklinger, and Athman Bouguettaya*

Constructing an Authentication Token to Access External Services in Service Aggregation .................. 321
*Peishun Wang, Yi Mu, Willy Susilo, and Jun Yan*
Application and Industry Session 5—Semantic Web and Pattern

Discovering Hierarchical Patterns in Event-Based Systems ................................................................. 329
  Hannes Obweger, Josef Schiefer, Peter Kepplinger, and Martin Suntinger

Alternative Approaches for Workflow Similarity .................................................................................. 337
  Andreas Wombacher and Chen Li

A Knowledge Acquisition Method for Improving Data Quality in Services Engagements .................. 346

Application and Industry Session 6—QoS Enforcement

Feedback Controlled Quality of Service Enforcement for Service Oriented Architectures ..................... 354
  Andreas Lodde, Antoine Schlechter, Pascal Bauler, Nicolas Biri, and Fernand Feltz

Defining Metrics for Loose Coupling Evaluation in Service Composition ........................................... 362
  Anthony Hock-koon and Mourad Oussalah

Market-Based QoS Control for Voluntary Services ............................................................................. 370
  Yohei Murakami, Naoki Miyata, and Toru Ishida

Application and Industry Session 7—Business Process Integration and Management

Mobile Process Management through Web Services ............................................................................. 378
  Massimiliano de Leoni and Massimo Mecella

Towards Green Business Process Management .................................................................................... 386
  Konstantin Hoesch-Klohe, Aditya Ghose, and Lam-Son Lê

Process Behavior Analysis: Mechanism to Enable Superior Customer Experience ............................... 394
  Karthik Visweswariah, Om D. Deshmukh, Ashish Verma, Sai Zeng, Chitra Dorai, Maureen Rzasa, and Gary Wright

Application and Industry Session 8—Business Driven Services Composition

A Cost-Minimizing Service Composition Selection Algorithm Supporting Time-Sensitive Discounts ................................................................. 402
  Lei Xu and Brendan Jennings

A Business Centric End-to-End Monitoring Approach for Service Composites ...................................... 409
  Geetika T. Lakshmanan, Paul Keyser, Aleksander Slominski, Francisco Curbera, and Rania Khalaf

Smarter Heuristics for Business-Driven Services Selection in Multi-services P2P Grids ......................... 417
  Alvaro Coêlho and Francisco Brasileiro
# Application and Industry Session 9—Software Tools for Services Computing

**AHA: Asset Harvester Assistant**

Debdoot Mukherjee, Senthil Mani, Vibha Singhal Sinha, Rema Ananthanarayanan, Biplav Srivastava, Pankaj Dhoolia, and Prahlad Chowdhury

- Page 425

**A Link Obfuscation Service to Detect Webbots**

Douglas Brewer, Kang Li, Laksmish Ramaswamy, and Calton Pu

- Page 433


Jin Zeng, Hailong Sun, Xudong Liu, Ting Deng, and Jianing Zou

- Page 441

# Application and Industry Session 10—Information Systems for Business Processes

**Prospective and Retrospective Provenance Collection in Scientific Workflow Environments**

Chunhyeok Lim, Shiyong Lu, Artem Chebotko, and Farshad Fotouhi

- Page 449

**Correcting Interaction Mismatches for Business Processes**

Shuai Gong, Jinhua Xiong, Zhiyong Liu, and Cheng Zhang

- Page 457

**Using Complex Event Processing for Dynamic Business Process Adaptation**

Gabriel Hermosillo, Lionel Seinturier, and Laurence Duchien

- Page 466

# Application and Industry Session 11—Service Platform and Middleware

**Message-Based Service Brokering and Dynamic Composition in the SAI Middleware**

Federica Paganelli, David Parlanti, and Dino Giuli

- Page 474

**An Assessment of Middleware Platforms for Accessing Remote Services**

Young-Woo Kwon, Eli Tilevich, and William R. Cook

- Page 482

**Extensible and General Service-Oriented Platform: Experience with the Service Abstract Machine**

Eric Simon, Jacky Estublier, and Diana Moreno-Garcia

- Page 490

# Application and Industry Session 12—Cloud Services

**Wukong: Toward a Cloud-Oriented File Service for Mobile Devices**

Huajian Mao, Nong Xiao, Weisong Shi, and Yutong Lu

- Page 498

**X-RIME: Cloud-Based Large Scale Social Network Analysis**

Wei Xue, JuWei Shi, and Bo Yang

- Page 506

**Online Self-Reconfiguration with Performance Guarantee for Energy-Efficient Large-Scale Cloud Computing Data Centers**

Haibo Mi, Huaimin Wang, Gang Yin, Yangfan Zhou, Dianxi Shi, and Lin Yuan

- Page 514
Application and Industry Session 13—Cloud Computing

Resource Allocation and SLA Determination for Large Data Processing Services over Cloud .................................................................................................................................................522

K. Hima Prasad, Tanveer A. Faruquie, L. Venkata Subramaniam, Mukesh Mohania, and Girish Venkatachaliah

Cross Enterprise Improvements Delivered via a Cloud Platform: A Game Changer for the Consumer Product and Retail Industry ........................................................................................................................................530

Trieu Chieu, Shubir Kapoor, Ajay Mohindra, and Anees Shaikh

PeopleCloud Service for Enterprise Crowdsourcing ..................................................................................................................................................538

Mariana Lopez, Maja Vukovic, and Jim Laredo

Application and Industry Session 14—SOA Management and Coordination

SOAC: A Conceptual Model for Managing Service-Oriented Authorization ........................................................................................................546

Haiyang Sun, Weiliang Zhao, and Jian Yang

Legacy Asset Analysis and Integration in Model-Driven SOA Solution ........................................................................................................................................554

Nianjun Zhou, Liang-Jie Zhang, Yi-Min Chee, and Lei Chen

Towards a Variability Model for SOA-Based Solutions ........................................................................................................................................562

Nanjangud C. Narendra and Karthikeyan Ponnalagu

Application and Industry Session 15—SOA and Business

Aligning IT Service Levels and Business Performance: A Case Study ..................................................................................................................570

Thiago Barroero, Giannario Motta, Giovanni Pignatelli, Mario Bochicchio, Antonella Longo, and Alessandra Raffone

Process as a Service—Distributed Multi-tenant Policy-Based Process Runtime Governance ........................................................................................................................................578

MingXue Wang, Kosala Yapa Bandara, and Claus Pahl

An Ontology to Support Context-Aware B2B Services ........................................................................................................................................586

P.S. Tan, A.E.S. Goh, and S.S.G. Lee

Business-Driven Trust Federation Management for Service Marketplaces ..................................................................................................594

Ji Hu, Xiaoyi Fan, and Robin Fischer

Application and Industry Paper—SCC 2009

Towards a Unified Service Description Language for the Internet of Services: Requirements and First Developments ........................................................................................................................................602

Jorge Cardoso, Alistair Barros, Norman May, and Uwe Kylau

Work-in-Progress Session 1—Implementation of Services Computing

Formal Passive Testing of Service-Oriented Systems ........................................................................................................................................610

César Andrés, M. Emilia Cambronero, and Manuel Núñez

A Service Identification Framework for Legacy System Migration into SOA ..................................................................................................614

Saad Alahmari, Ed Zaluska, and David De Roure
Towards a Foundation for Quantitative Service Analysis—An Approach from Business Process Models ..........................................................................................................................618
  Naveen Kulkarni, Deepi Parachuri, and Shashank Trivedi
Towards Self-Aware Performance and Resource Management in Modern Service-Oriented Systems ......................................................................................................................................621
  Samuel Kounev, Fabian Brosig, Nikolaus Huber, and Ralf Reussner

Work-in-Progress Session 2—Data Management in Services Computing
Supporting Geosciences Web Services Metadata Management and Discovery ...........................................625
  Pearl Brazier, Artem Chebotko, Eric Gonzalez, Andrey Kashlev, and Anthony Piazza
Utility Decomposition and Surplus Redistribution in Composite SLA Negotiation ..........................................................627
  Jan Richter, Mohan Baruwal Chhetri, Ryszard Kowalczyk, Bao Quoc Vo,
  Muhammed Adeel Talib, and Alan Colman
Functionality-Driven Clustering of Web Service Registries ........................................................................631
  Mohamed Sellami, Walid Gaaloul, and Samir Tata

Work-in-Progress Session 3—Management and Support of Services Computing
Toward Consistent Combination of Service Selection Methods and Process Adaptation Methods ..........................................................635
  Fuyuki Ishikawa
Task Exception Handling in the VIEW Scientific Workflow System ........................................................................637
  Dong Ruan and Shiyong Lu
A Framework for Shrink-Wrapping Security Services ...........................................................................639
  Gleneesha Johnson, Ashok Agrawala, and Elodie Billionniere
Enhancing Web Service Registries with Semantics and Context Information ........................................641
  Lu Chen, Yan Li, and Randy Chow

Work-in-Progress Session 4—Services Delivery and Applications
Converged Service Delivery: Modeling and Engineering ..............................................................................645
  Noëmie Simoni, Chunyang Yin, and Philippe Coudé
A Framework for Personalized Context-Aware Search of Ontology-Based Tagged Data ..........................................................649
  Knarig Arabshian
A Novel Software Environment for Developing Migrating Internet Applications Based on Fusion of Mobile Agent, Web Services and BPEL Technologies ........................................................................................................651
  J.M. Gnanasekar and Venkatesan D. Pillai

Author Index .......................................................................................................................................................653
A Model for Visual Specification of e-Contracts

Enrique Martínez, Gregorio Díaz, M. Emilia Cambronero

Department of Computer Science
University of Castilla - La Mancha
Albacete, Spain
{emartinez, gregorio, emicp}@dsi.uclm.es

Gerardo Schneider
Department of Applied IT
University of Gothenburg, Sweden
Department of Informatics
University of Oslo, Norway
gersch@chalmers.se

Abstract—In a web service composition, an electronic contract (e-contract) regulates how the services participating in the composition should behave, including the restrictions that these services must fulfill, such as real-time constraints. In this work we present a visual model that allows us to specify e-contracts in a user friendly way, including conditional behavior and real-time constraints. A case study is presented to illustrate how this visual model defines e-contracts and a preliminary evaluation of the model is also done.

Keywords—contracts; deontic specifications; visual models;

I. INTRODUCTION

Most of the research efforts spent on the theory of electronic contracts in service-oriented architectures have been oriented to the formal definition of a public service interface with which other services can interact [5]. However, services e-contracts not only refer to the interfaces provided by these services, they also refer to a certain number of clauses that must be satisfied by several parties. These clauses regulate how participants should behave, what are the penalties in case of misbehavior, and under which conditions such clauses must be enacted (e.g. time restrictions such as deadlines). When a clause is violated, the contract is breached. However, if the clause defines a reparation (secondary clauses that come into force when the main clause is not satisfied), and this reparation is fulfilled, then the clause is eventually fulfilled [1].

Recently some works about specifying services e-contracts in a formal manner have been released [2], [4], [7], [18]. These approaches consist of formal languages which are hard to study and manipulate by untrained final users of this technology, as business process developers.

The goal of this work is to introduce a new approach for the specification of e-contracts in a user friendly way. E-contracts may be complex, consisting of composite clauses making reference to other clauses in the same or in another contract. Furthermore, we consider contracts with timed restrictions and conditions under which the contract clauses must be applied. Hence, our approach is based on a visual model, since it is well-known that the use of visual models makes easier the perception of knowledge, and in this way, the intuitive understanding, reading and maintenance of complex problems [8], [9]. This approach can be useful not only in service-oriented architectures but also in component-based systems, requirements acquisition, software product lines, etc.

The contribution of this work is twofold. First, we define a visual model to deal with the acquisition and elicitation of requirement and restrictions. This visual model allows us to specify the notions of obligation, permission and prohibition [14] as elements of a hierarchical diagram. In this way, these elements are clauses that can be refined hierarchically and can include (real-time) constraints and a reparation that must be performed when the main norm is not fulfilled. Second, a preliminary evaluation of the model is presented, based on user-based tests and the principles defined in [16].

The rest of the work is structured as follows: Section II describes the visual model we have developed, showing in Section III a case study where this model is applied. In Section IV we present the results of the preliminary evaluation and Section V is concerned with related work. Finally, in Section VI, we present the conclusions and future work.

II. VISUAL MODEL

In our visual model we define a hierarchical tree diagram used to specify the contract clauses. We call this diagram Contract-Oriented Diagram or C-O Diagram for short.

In Figure 1 we show the basic element of our C-O Diagram. It corresponds to a contract clause and we call it box. This box consists of four fields, allowing us to specify normative aspects or simple norms (P), reparations (R), conditions (g) and time restrictions (tr). Each box has a name and an agent. The name is useful both to describe the clause and to reference the box from other clauses, so it must be unique. The agent indicates who is the performer of the action.

On the left-hand side of the box we specify the conditions and restrictions. The guard g specifies the conditions under which the contract clause must be taken into account. The
The propositional content $P$, on the center, is the main field of the box, and it is used to specify normative aspects (obligations, permissions and prohibitions) that are applied over actions, and/or the actions themselves.

The last field of these boxes, on the right-hand side, is the reparation $R$. This reparation, if specified by the contract clause, is another contract that must be satisfied in case the main norm is not satisfied, considering the clause eventually satisfied if this reparation is satisfied.

These basic elements of a C-O Diagram can be refined by using AND/OR refinements, as shown in Figure 2, in the same way that we refine goals into subgoals in goal model diagrams [21]. It is also possible to use another refinement to specify a temporal relationship of sequence (SEQ) between subclauses, as shown in the left part of Figure 3. The aim of these refinements is to capture the hierarchical clause structure followed by most contracts. An AND-refinement means that all the subclauses must be satisfied in order to satisfy the parent clause. An OR-refinement means that it is only necessary to satisfy one of the subclauses in order to satisfy the parent clause, so as soon as one of its subclauses is fulfilled, we conclude that the parent clause is fulfilled as well. A SEQ-refinement means that the norm specified in the target box (SubClause2 in Figure 3) must be fulfilled after satisfying the norm specified in the source box (SubClause1 in Figure 3). In this way, we can build a hierarchical tree with the clauses defined by the contract, where the leaf clauses correspond to the atomic clauses, that is, to the clauses that cannot be divided into subclauses.

Finally, there is another structure that can be used to model repetition, apart from the refinements previously defined. This structure is represented as an arrow going from a subclause to one of its ancestor clauses (or to itself), meaning the repetitive application of all the subclauses of the target clause after satisfying the source subclause. For example, in the right part of Figure 3, we have an OR-refinement with an arrow going from SubClause1 to Clause. It means that after satisfying SubClause1 we apply Clause again, but not after satisfying SubClause2.

In the next paragraphs we describe each one of the fields of a box in more detail.

**Propositional Content $P$:** This is the main field of a box. It allows us to specify the obligations, permissions and prohibitions, as defined in deontic logic [15], that the contract must be satisfied. In this work, we follow an ought-to-do approach [22], i.e., these normative aspects are applied over actions performed by the participants in the contract.

Although we will see later that it is possible to specify compound actions, we only consider the specification of atomic actions in the $P$ field of the leaf clauses of our diagrams. These actions are denoted by lower case Latin letters ("a", "b", "c", \ldots). We use a dash ("-"") to denote that there is no action specified in the no leaf clauses.

The composition of actions can be achieved by means of the different kinds of refinement. In this way, an AND-refinement can be used to model concurrency "&" between actions, an OR-refinement can be used to model a choice "+" between actions, and a SEQ-refinement can be used to model sequence ";" of actions. In Figure 4 we can see an example about how to model these compound actions through refinements, given two atomic actions $a$ and $b$.

The deontic norms (obligations, permissions and prohibitions) that are applied over these actions can be specified in any clause of our C-O Diagrams, affecting all the actions in the leaf clauses that are subclauses of this clause. If it is the case that the clause where we specify the deontic norm is a leaf clause, the norm only affects the atomic action we have in this clause. We use an upper case "O" to denote an obligation, an upper case "P" to denote a permission, and an upper case "F" to denote a prohibition (forbidden). These letters are written in the top left corner of field $P$.

The composition of deontic norms is also achieved by means of the different refinements we have in C-O Diagrams. Thus, an AND-refinement corresponds to the conjunction operator "\&" between norms, an OR-refinement corresponds to the choice operator "+" between norms, and a SEQ-refinement corresponds to the sequence operator ";" between norms. For example, we can imagine having a leaf clause specifying the obligation of performing an action $a$, written as $O(a)$, and another leaf clause specifying the obligation of performing an action $b$, written as $O(b)$. These two norms can be combined in the three different ways mentioned before through the different kinds of refinement (Figure 5).

However, the specification of obligations, permissions and prohibitions in our diagrams must fulfill the following rules:

1. At least one deontic norm must be specified in each one of the branches of our hierarchical tree of clauses, i.e., we cannot have an action without a deontic norm applied over it.
2. No more than one deontic norm can appear in each one of the branches of our hierarchical tree of clauses, i.e., we cannot have deontic norms applied over other deontic norms.
3. The deontic norms we take into account to check restrictions 1. and 2. can be shared by several branches, i.e., when we have a deontic norm applied over a compound action, this norm is part of several branches of our diagram.

The repetition of both, actions and deontic norms, can be achieved by means of the repetition structure we define in C-O Diagrams. The meaning of this structure is similar to the Kleene’s star operator “∗” [10] applied over the elements of the target clause of the arrow, but it is richer in the sense that the repetition can be conditioned to the satisfaction of the source clause of the arrow and not other alternative clause.

**Reparation R:** This field of a box can state a new contract that must be satisfied when the main field P is not satisfied (a prohibition is violated or an obligation is not fulfilled, there is not reparation for permission). This new contract can be just a new norm, but it can also be a new hierarchical tree of clauses, including their own reparations. In this way, we are able to specify nested reparations in our C-O Diagrams.

The field R is only allowed in the clauses of our diagrams where we specify a deontic norm of obligation or prohibition in field P, being forbidden in the other clauses. E.g., we can imagine a main contract C stating that we have the obligation of performing an atomic action a and the prohibition of performing an atomic action b. However, if we do not perform the obligatory action a, we can compensate it by fulfilling another contract called C₁, consisting of performing an action c or an action d, and if we perform the forbidden action b, we can compensate it just by performing an action e. This situation can be modeled in our diagrams as shown in Figure 6.

At this point we can see clearly the difference between having a composition of obligations over atomic actions and having an obligation over a compound action. While the former allows us to specify a different reparation for each one of the atomic actions we are obliged to do, the latter only allows us to specify one reparation for the compound action that is under the obligation operator. For the first case, we can consider the diagrams we have in Figure 5, where it is possible to specify a different reparation in each one of the leaf clauses of the diagrams. For the second case, we can imagine having the diagrams shown in Figure 7, where we can only specify reparations in the no leaf clauses where we have the obligations, affecting these reparations the whole composition of actions.

This difference is a bit trickier if we consider prohibitions or permissions, because it not only concerns to the specification of reparations [17]. For example, given two atomic actions a and b, the meaning of prohibiting the sequence of these two actions, written as $F(a,b)$, is different from the meaning of prohibiting action a, and next prohibiting action b, written as $F(a).F(b)$. In the first case, the sequence of actions starting with a and continuing with any action different from b is allowed, while in the second case any sequence of actions starting with a is forbidden. Similar distinctions exist when we consider permissions instead of prohibitions.

**Guard g:** This field of a box is a boolean expression that evaluates some information provided by the clause specification, telling us under which conditions the clause must be taken into account. E.g., in a car insurance contract we can have a clause that is only applied to people under the age of 21. In that case, we must include in the box modeling this clause a guard like $\text{age} < 21$.

Basically, a guard is a set of expressions that evaluate to a boolean (true or false) combined by means of conjunctions (and), disjunctions (or), and negations (not). These expressions can include constant values, variables, and equality and inequality operators ($==$, $!=$, $<$, $>$, …).

When the guard condition corresponding to a subclause of
an **AND**/**SEQ** refinement evaluates to `false`, the subclause is trivially satisfied, so we only must check the subclauses with a `true` guard (or without guard). However, when the guard condition corresponding to a subclause of an **OR**-refinement evaluates to `false`, we cannot satisfy that subclause in order to satisfy the parent clause, so it is necessary to satisfy one of the other subclauses with a `true` guard (or without guard). E.g., in a payment system we can have the obligation of paying by cash or by credit card, but the second option is conditioned to be of legal age, so in case it is not satisfied we have only the possibility of paying by cash.

**Time restriction** `tr`: Each clause can have associated a time restriction, e.g., deadlines, timeouts, etc. These real-time aspects are expressed in the boxes of our diagram by means of **intervals** within the field `tr`. These intervals indicate the period of time in which the clauses must be satisfied.

The time restrictions can be specified in two different ways within the boxes: we can specify the **dates** binding the beginning and the end of the time frame corresponding to the clause (**absolute time**), or we can specify a deadline saying the number of **time units** that can elapse before the clause is satisfy from the moment at which another clause is satisfied or from the moment at which the contract comes into effect (**relative time**). For example, in the case of absolute time, we can model a contract stating that a clause must be satisfied in the **first five days of October** (interval [10/1 00:00, 10/5 23:59]), whereas in the case of relative time a contract can state that a clause must be satisfied not later than **five days after satisfying another clause C** (interval [C, 5days]).

When we have a time restriction specified in a clause that is refined into subclauses, this restriction affects all the subclauses, i.e., all the subclauses necessary to satisfy the parent clause must be satisfied in the time frame specified in this parent clause. Otherwise, the parent clause is considered unfulfilled.

### III. Case Study: A Software Provision System

In this case study we present a contract regulated composition of services between a client and a software provider. The case study is inspired by the one published in [13], but with some modifications in the composition and the contract specification, including the definition of real-time constraints. The parties involved in this contract are the **client**, the **software provider**, and the **testing agency**.

The scenario we are considering in this case is the following: everything starts when the **client** asks the **software provider** to develop a new software. The **software provider** develops the different components needed to implement this new software. The **software provider** informs twice the **client** about the progress of the software development. If the **client** wants any changes in the software, he can request them after the first update. Any changes suggested after the second update are considered a violation. The **client** can recover from this violation by paying a penalty to the **software provider** or by withdrawing the suggested changes. Every update is followed by a payment from the **client** to the **software provider**. If the **software provider** does not send the updates to the **client** at the schedule time (three months for the first update and nine months for the second update), it is considered a violation of the contract that is repaired by paying a penalty charge to the **client**.

Once all the software components are implemented, the **software provider** integrates these components and sends the final product to the **testing agency** for testing. Then this **testing agency** sends testing reports to the other parties. If the tests fail, the components are revised by the **software provider** and then tested again. Finally, if the tests succeed, the **software provider** delivers the final product to the **client**.

In Table I we show a list of the obligations, permissions and prohibitions we can deduce from the scenario described above. We can see there are two clauses specifying real-time constraints, **Clause 1** and **Clause 4**. If we consider the moment when the **client** asks the **software provider** to develop a new software as the moment when the contract comes into effect (denoted as `0`), the time **software provider** has to update **client** for the first time after being asked to develop the new software is three months (denoted as `3M`) according to **Clause 1**. Moreover, from the above scenario we can deduce that there is a second time constraint, that the time **software provider** has to update **client** for the second time after updating him for the first time is nine months (denoted as `9M`) according to **Clause 4**. We can also see in Table I that **Clause 9** and **Clause 10** are conditioned to the result of the tests, so we consider a boolean variable we call `Tests_Ok` specifying if the tests have succeeded.

The main problem we have with this textual specification of the contract is that it is no clear the relationship existing between the different clauses, which makes difficult any kind of analysis of the contract. Therefore, we aim at a specification language that clearly defines the relationship between the different clauses, allowing the analysis of the contract, but not so formal that an expert is needed. C-O **Diagrams** satisfies the above requirements. In what follows
we explain how to model the contract with these diagrams, taking into account the information provided in Table I. We use $a_n$ to denote the action performed by clause number $n$ and $r_m$ to denote the reparation defined for clause number $m$.

In Figure 8 we show the top-level of the C-O Diagram we specify for the contract, where we have grouped the clauses in Table I into three more general clauses with a sequence relationship between them: First_Update (Clause 1, Clause 2 and Clause 3), Second_Update (Clause 4, Clause 5 and Clause 6), and Correct_Software (Clause 7, Clause 8, Clause 9 and Clause 10). These three clauses cover the three different phases we can distinguish in the contract.

The decomposition of clause First_Update into subclauses can be seen in Figure 9. In this case, we first have the specification of the obligation contained in Clause 1, including the deadline (three months from the beginning) and the reparation, and after that we have the specification of Clause 2 and Clause 3, both affecting the behavior of the client and with an $\land$ relation between them, composing the parent clause Client_First_Behavior.

The decomposition of clause Second_Update into subclauses can be seen in Figure 10 and it is very similar to the previous one. We first have the specification of the obligation contained in Clause 4, including the deadline (nine months from the fulfillment of Clause 1, abridged as $C_1$) and the reparation, and after that we have the specification of Clause 5 and Clause 6, both affecting the behavior of the client and with an $\land$ relation between them, composing the parent clause Client_Second_Behavior. The main difference is that the client is not allowed to request any change, so instead
of a permission we have now a prohibition in Clause 6. This prohibition specifies two possible reparations, so we use \( r_{6a} \) to represent the payment of a penalty and \( r_{6b} \) to represent the withdrawal of the changes by the client. We write this complex reparation in the same diagram instead of referencing another diagram just to save space.

Finally, the clause Correct Software is decomposed as shown in Figure 11. We first have the obligation contained in Clause 7, about sending the software to the testing agency. After that we have the obligation contained in Clause 8, about sending the reports to the software provider and to the client. These two actions can be performed concurrently, so we call \( s_{8a} \) the action of sending the testing reports to the software provider and \( s_{8b} \) the action of sending these reports to the client. Last, we have the specification of Clause 9 and Clause 10, both affecting the software provider and with a + relation between them, composing the parent clause Test Result. We notice that the selection of one subclause or the other is related to the result of the tests, so we use the variable Test_OK to model that situation. Each time we apply Clause 9 (software provider revises the components), we repeat the sequence of software testing, going back to the application of Clause 7. If Clause 10 is applied, the software provider delivers the final product to the client and the contract finishes.

IV. Evaluation of the Model

In this section we present an evaluation of the visual model described above, divided into a qualitative and a quantitative evaluation. For the former evaluation we discuss how the model fits some of the most important principles for designing effective visual notations defined in [16]. First, the principle of semiotic clarity is accomplished by the model, as there is only one graphical structure corresponding to each semantic concept and vice versa. Second, the principle of perceptual discriminability is taken into account to differentiate between refinements, having each kind of refinement a clearly distinct shape and using the text with the name of the refinement to complement the graphics (principle of dual coding). We also have that the number of different graphical symbols in the model is under the upper limit of six categories for graphics complexity, so the principle of graphic economy is accomplished. Finally, the principle of complexity management is covered by the modularization of the diagrams, as we have done in the case study. As we have seen in the previous section, modules are combined by having the same box appearing in several diagrams (principle of cognitive integration).

The quantitative evaluation of the model is done by means of user-based tests. Our purpose with these tests is to compare C-O Diagrams with textual notations for e-contracts, so in the tests we use a textual version of the diagrams very similar to CL language [18], as it is closely related to the approach followed by C-O Diagrams, but adding agents and time aspects that are not currently supported by CL. We have designed two tests to compare the understandability of both, textual and visual representation, where we have that the same semantic concept is represented by a C-O Diagram in one test and by the textual language in the other test, asking the same questions in both cases. These tests have been done by 20 students of our university who have previously attended two lectures about e-contracts and C-O Diagrams, and we have obtained the results shown in Table II. In the table we show the average error rate and time taken by students in both cases, for textual representation and for visual representation. The results are divided into four rows depending on the kind of contracts that the questions correspond to: contracts with only a basic deontic norm, contracts with a composition of deontic norms, contracts with temporal restrictions, and contracts including reparations. As we can see, in the simplest case there is no difference between textual and visual representation, but in all the other cases we obtain better results for visual representation than for textual representation.

<table>
<thead>
<tr>
<th>Composition</th>
<th>Time (sec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Textual</td>
<td>Visual</td>
</tr>
<tr>
<td>Basic</td>
<td>1.8%</td>
</tr>
<tr>
<td>Composition</td>
<td>22.0%</td>
</tr>
<tr>
<td>Temporal</td>
<td>12.1%</td>
</tr>
<tr>
<td>Reparation</td>
<td>54.5%</td>
</tr>
</tbody>
</table>

Table II

<table>
<thead>
<tr>
<th>Composition</th>
<th>Time (sec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Textual</td>
<td>Visual</td>
</tr>
<tr>
<td>Basic</td>
<td>1.8%</td>
</tr>
<tr>
<td>Composition</td>
<td>22.0%</td>
</tr>
<tr>
<td>Temporal</td>
<td>12.1%</td>
</tr>
<tr>
<td>Reparation</td>
<td>54.5%</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Composition</th>
<th>Time (sec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Textual</td>
<td>Visual</td>
</tr>
<tr>
<td>Basic</td>
<td>1.8%</td>
</tr>
<tr>
<td>Composition</td>
<td>22.0%</td>
</tr>
<tr>
<td>Temporal</td>
<td>12.1%</td>
</tr>
<tr>
<td>Reparation</td>
<td>54.5%</td>
</tr>
</tbody>
</table>
this scale for our model. This is a score that clearly shows
the user’s preference to use the visual model instead of the
textual notation.

All these tests can be accessed via the Moodle course “C-
O Diagrams” in http://moodle.retics.uclm.info/. Anyone can
access this course login as a guest. The tests are available
at the Social Activities box, through the preview option.

V. RELATED WORK

To the best of our knowledge, there is not any other visual
model specially created for the definition of e-contracts.
However, several works in the literature define a meta-
model for the specification of e-contracts whose purpose
is the enactment or the enforcement of this e-contract.
For instance, in [6] Chiu et al. present a meta-model for
e-contract templates written in UML, where a template
consists of a set of contract clauses of three different types:
obligations, permissions and prohibitions. These clauses are
later mapped into event-condition-action (ECA) rules for
contract enforcement purposes, but the templates do not
include any kind of reparation or recovery associated to the
clauses, and the way of specifying the different possible rela-
propose another meta-model of e-contracts based on entity-
relationship diagrams that they use to generate workflows
supporting e-contract enactment. This meta-model includes
clauses, activities, parties and the possibility of specifying
exceptional behavior, but this approach is not based on the
deontic notions of obligation, permission and prohibition,
and says nothing about including real-time aspects natively.
Another approach can be found in [20], where Rouached
et al. propose a contract layered model for modeling and
monitoring e-contracts. This model consists of a business
entities layer, a business actions layer, and a business rules
layer. These three layers specify the parties, the actions
and the clauses of the contract respectively, including the
conditions under which these clauses are executed. However,
real-time restrictions are not included and the specification
of the clauses follows an operational approach, not a deontic
approach.

The approach followed in C-O Diagrams for the specifi-
cation of e-contracts is close related to the formal language
\( \mathcal{CL} \) [18]. In this language a contract is also expressed as
a composition of obligations, permissions and prohibitions
over actions, and the way of specifying reparations is the
same that in our visual model. The main difference with C-
O Diagrams is that \( \mathcal{CL} \) does not support the specification
of agents nor timing constraints natively, so they have to be
encoded in the definition of the actions. Also, in \( \mathcal{CL} \) there
is no sequence operator to combine the different clauses,
so the notion of sequence has to be expressed always
by means of specifying the application of a clause after
performing a certain action (denoted as \( [\alpha]C \), where \( \alpha \)
is a compound action and \( C \) is a general contract clause),
like in propositional dynamic logic. Refer to [15] for a general
description of deontic logic.

In [14] Marjanovic and Milosevic also defend a deontic
approach for formal modeling of e-contracts, paying special
attention to the modeling of time aspects. They distinguish
between three different kinds of time in e-contracts: absolute
time, relative time and repetitive time. The two first kinds
are supported by C-O Diagrams, but repetitive time is not
included yet in our model. Nevertheless, with the combination
of the other two kinds of time and the repetition structure, we
can achieve some repetitive time behaviors in our model. In
[13] Lomuscio et al. present an approach to verify contract-
regulated service compositions. They use the orchestration
language WS-BPEL to specify all the possible behaviors of
each service and the contractually correct behaviors. After
that, they translate these specifications into timed automata
supported by the MCMAS model checker to verify the
behaviors automatically. In this work we have that the scope
of the e-contracts is limited to web services compositions,
specifying the e-contract corresponding to each one of the
services separately. The specification of real-time constraints
is not allowed because they are not supported by MCMAS
and the deontic norms are restricted to only obligations.

VI. CONCLUSIONS AND FUTURE WORK

In this paper we have presented C-O Diagrams, a new
visual formalism for electronic contracts. Though we have
shown the applicability of such diagrams on a case study taken from SOA, their usefulness go beyond services. They can be used for negotiation via email, contracting on the Internet, performing an Electronic Data Interchange Agreement (EDI), just to mention a few. Indeed, C-O Diagrams may also be used as an intermediate (formal) language for other applications. We are currently working on their use in the context of requirements engineering (to formalize requirements), product software families (as an extension of feature diagrams), and as a visual representation of CL contracts.

We are also working on a formal semantics based on automata, making it possible to formally analyze C-O Diagrams using for instance UPPAAL [12]. Besides the application of C-O Diagrams to the mentioned domains, we intend to explore how we can automatically obtain diagrams from a controlled (structured) English, by using for instance the Grammatical Framework (GF) [19]. We also envisage the possibility of specifying a different diagram for each one of the parties involved in an e-contract instead of having a global C-O Diagram with multiple agents. This compositional approach can be useful if we define a composition operator, specifying when two of these new C-O Diagrams can be composed and the result of the composition.

ACKNOWLEDGMENT

Partially supported by the Spanish government (co-financed by FEDER founds) with the project TIN2009-14312-C02-02, the JCCLM regional project PETH2009-0232-7745, and the Nordunet3 project “COSoDIS”. The first author is supported by the European Social Fund and the JCCLM.

REFERENCES


