Inf Syst Front (2013) 15:167–192
DOI 10.1007/s10796-012-9365-x

Semantic information and knowledge integration through argumentative reasoning to support intelligent decision making

Naeem Khalid Janjua · Farookh Khadeer Hussain · Omar Khadeer Hussain

Published online: 19 July 2012
© Springer Science+Business Media, LLC 2012

Abstract The availability of integrated, high quality information is a pre-requisite for a decision support system (DSS) to aid in the decision-making process. The introduction of semantic web ensures the seamless integration of information derived from diverse sources and transforms the DSS to an adoptable and flexible Semantic Web-DSS (Web-DSS). However, due to the monotonic nature of the layered development of semantic web, it lacks the capability to represent, reason and integrate incomplete and conflicting information. This, in turn, renders an enterprise incapable of *knowledge integration*; that is, integration of information about a subject that could potentially be incomplete, inconsistent and distributed among different Web-DSS within or across enterprises. In this article, we address the issues of incomplete and inconsistent semantic information and knowledge integration by using argumentation and argumentation schemes. We discuss the Argumentation-enabled Information Integration Web-DSS (Web@IDSS) along with its syntax and semantics for semantic information integration, and devise a methodology for sharing the results of Web@IDSS in Argument Interchange Format (AIF) format. We also discuss Argumentation-enabled Knowledge Integration Web-DSS (Web@KIDSS) for semantic knowledge integration. We provide formal syntax and semantics for the Web@KIDSS, propose a conceptual framework, and describe it in detail. We present the algorithms for knowledge integration and the prototype application for validation of results.

Keywords Semantic web · Information integration · Argumentation · Argumentation schemes · Web based DSS

1 Introduction

Decision support systems (DSS) are a broad category of interactive computer-based information systems designed and developed over the last forty years for a wide range of domains with the objective of constructing the reasons with which a decision maker will convince himself and other actors involved in the decision-making process (Power 2002; Power and Sharda 2009; Shim et al. 2002). If we examine the research conducted over the past few years, the synergies among technologies of the Internet, World Wide Web (WWW) and Artificial Intelligence (AI) have enabled the single user-focused DSS to evolve into an intelligent and complex Web-based DSS (Web-DSS) (March and Hevner 2007). The current Web-DSS are compatible with new technologies for business intelligence and provide a more transparent interaction between system and decision maker to improve the efficiency and

N. K. Janjua · O. K. Hussain
School of Information Systems, Curtin Business School,
Curtin University, Perth, WA, Australia

N. K. Janjua
e-mail: naeem.janjua@gmail.com

O. K. Hussain
e-mail: o.hussain@cbs.curtin.edu.au

F. K. Hussain (✉)
Decision Support and e-Service Intelligence (DeSI) Lab,
Quantum Computation and Intelligent Systems (QCIS),
School of Software, Faculty of Engineering and Information
Technology, University of Technology Sydney, Ultimo,
NSW 2007, Australia
e-mail: Farookh.Hussain@uts.edu.au

effectiveness of intelligent decision making (Liu et al. 2010).

With the current proliferation and widespread adoption of e-business, manufacturing and business are extending beyond enterprise boundaries (Norta and Eshuis 2010; Alaranta and Henningsson 2008). Moreover, enterprises are involved in collaboration and mergers with other enterprises on a global scale. This has created a demand for Intelligent Information Integration (III) to integrate, on demand, information from heterogeneous data sources, applications and environments, in order to facilitate inter-operation and collaboration, and provide intelligent decision support through Web-DSS, leading to a more efficient decision-making process at enterprise level (March and Hevner 2007; Seng and Kong 2009; Osei-Bryson and Ngwenyama 2008). It is important to note here that when we refer to 'information' we mean the following:

- the business polices or rules governing the inference mechanism, and
- the data to which the inference mechanism is being applied.

The semantic web, an extension of the current World Wide Web (WWW), is a step towards addressing the aforementioned challenge, i.e. Intelligent Information Integration (III), pertinent to Web-based DSS. The semantic web is seeking a universal medium for data exchange, i.e. classifying, packaging and semantically enriching information for support of data automation, integration, and reuse across various applications (Suguri et al. 2008; Torroni et al. 2009). The core of the semantic web, i.e. ontologies, meta-data and relations for performing inference with rules, is a source of seamless information integration of heterogeneous information sources. However, the initially proposed single stack architecture (SSA) of semantic web by Tim Berners-Lee assumed that the semantic web stack is composed of a main language and every new development should be built on top of existing layers (Berners-Lee 2000; Lee 2003). In response to criticisms that this proposal is unrealistic and unsustainable, Berners-Lee then proposed an alternative multi-stack architecture (MSA) to overcome the limitation of SSA (Lee 2005, 2006). Although the MSA is more realistic in the long run, it provides a basis for only monotonic logic-based knowledge representation and reasoning; it does not provide a basis for representing, reasoning and integrating incomplete and/or contradictory information. In monotonic logic, once a conclusion has been drawn using currently available information, the availability of new information may negate or invalidate existing information. This is a significant problem with monotonic logics and requires re-examination of the entire reasoning cycles when new information, especially that which contradicts existing knowledge, is added to the knowledge base (Hurt 1998). Hence, the current generation of Web-DSS is not able to represent, reason and integrate incomplete and inconsistent information, irrespective of whether this information emerges from within the organization or outside it (Antoniou et al. 2004; Carlsson and Turban 2002). There has been much discussion in the literature on the development of defeasible or non-monotonic logic based Web-DSS systems that can integrate incomplete and conflicting information through reasoning and defining priorities among conflicting rules at compile time (Antoniou and Bikakis 2007; Bassiliades et al. 2004; Grosof et al. 2002). Reasoning is called 'defeasible' or 'non-monotonic' if a rule that supports a conclusion can be defeated by new information. However, such Web-DSS systems have two main limitations:

- They provide a formalism to represent and handle only individual preferences in the form of priorities in order to handle incomplete and conflicting information. However, Web-DSS are subject to inconsistencies deriving from multiple data/information sources and multiple users; therefore, it is not possible to define priorities in advance in order to resolve conflicts among business rules derived from multiple sources/users.
- The use of these priorities is usually embedded in the derivation mechanism and competing rules are compared individually during the derivation process. Therefore, the derivation notion is bound to one single comparison criterion. In such a scenario, the explanation of the results is based on a single criterion only and fails to take into account the multiple factors important for decision-making.

To overcome the limitations of current Web-DSS, we propose and develop a framework for semantic information and knowledge integration at the enterprise level for intelligent decision support. Figure 1 depicts a layered architecture for Intelligent Information Integration (III) spanning the enterprise. It starts with the semantic information integration phase followed by the semantic knowledge integration phase. Let us discuss each phase of III along with the contributions of this article.

1. **Semantic Information Integration (SII)**
   We define SII as the integration of semantic information elicited in the form of business rules and data, residing in different Web-based decision support systems. Our previous work focused on the de-

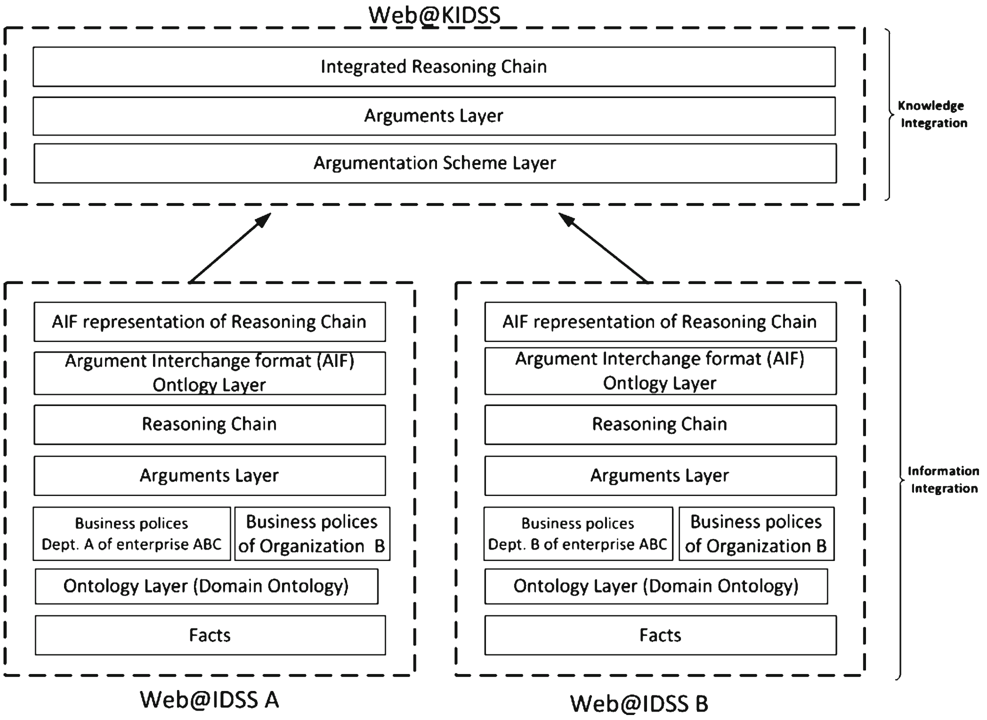

Web@KIDSS
Integrated Reasoning Chain
Arguments Layer
Argumentation Scheme Layer
Knowledge Integration
AIF representation of Reasoning Chain
Argument Interchange format (AIF) Ontlogy Layer
Reasoning Chain
Arguments Layer
Business polices Dept. A of enterprise ABC
Business polices of Organization B
Ontology Layer (Domain Ontology)
Facts
AIF representation of Reasoning Chain
Argument Interchange format (AIF) Ontlogy Layer
Reasoning Chain
Arguments Layer
Business polices Dept. B of enterprise ABC
Business polices of Organization B
Ontology Layer (Domain Ontology)
Facts
Information Integration
Web@IDSS A
Web@IDSS B

Fig. 1 Layered-view architecture for intelligent information integration (III)

sign and development of an argumentation-enabled intelligent decision support system (Web@IDSS) capable of providing decision support by integrating semantic information which potentially could be incomplete and inconsistent. Sections 4 and 5 provide detailed information about Argumentation and Web@IDSS respectively.
In this article, we extend the functionality of Web@IDSS with AIF reification in order to share the results of Web@IDSS over the enterprise intranet using Argument Interchange Format (AIF) ontology.

2. **Semantic Knowledge Integration(SKI)**
We define SKI as the integration of decisions or results about a subject that are potentially incomplete, inconsistent and distributed amongst different Web-DSS. To integrate the results of different Web-DSS published over the web or enterprise intranet in AIF compliant format, we propose and demonstrate extensions to Web@IDSS to address the issue of knowledge integration at enterprise level with the help of argumentation schemes known as Web-based Intelligent Knowledge Integration DSS (Web@KIDSS). The contributions of this paper to the existing literature body in terms of SKI are as follows:

(a) We propose and demonstrate the integration of different AIF-compatible reasoning chains using a user-defined argumentation scheme.
(b) We provide formal syntax and semantics for argumentative reasoning for semantic knowledge integration.
(c) We also discuss the system architecture in detail followed by algorithms used for semantic knowledge integration.
(d) We illustrate the use of Web@KIDSS with a case study, discuss prototype development, and indicate future directions.

The rest of the paper is structured as follows: Section 2 presents a review of the literature, elaborating and focusing on Semantic Web-DSS and Defeasible logic-based implementations of Web-DSS along with their limitations. Section 3 describes a case study related to information and knowledge integration in an enterprise. Section 4 discusses argumentation and argumentation schemes. Section 5 introduces semantic information integration using Web@IDSS. It describes the working of We@IDSS and AIF reification. This is followed by Section 6 which elaborates upon the knowledge integration process using Web@KIDSS. In this section, we define the formal syntax and semantics for knowledge integration. This section also elaborates on the proposed conceptual framework for semantic knowledge integration and discusses the algorithms in detail. This section is followed by implementation and prototype development in Section 7. Section 8 presents the conclusion and future directions.

2 Review of literature

In an open computing environment, such as the World Wide Web or an enterprise intranet, various decision support systems are expected to work together to support information exchange, processing, and integration. However, DSS are usually built by different people, at different times, to fulfil different requirements and goals leading to (Xue et al. 2009):

1. different supporting infrastructures
2. different syntactic representations of information
3. different schematic designs of information models
4. different semantics of information models.
5. conflicts among information, and the presence of incomplete information hinders its integration into information systems and afterwards knowledge integration at enterprise level.

Mostly, integration efforts have focused largely on the first four issues (Zhou et al. 2010; Seng and Kong 2009; Nguyen et al. 2011). In this paper, we discuss the fifth issue which has received little attention in existing literature: semantic information and knowledge integration in the presence of incomplete and conflicting information.

Recently, knowledge engineers have realized that they need to agree on a shared conceptualization of an application domain, known as an 'ontology' when developing two or more information systems which are syntactically and semantically interoperable (Muthaiyah and Kerschberg 2007). A number of researchers are working on different aspects of ontologies such ontology construction (Kim et al. 2011), ontology mapping (Chua and Goh 2010), ontology tailoring (Flahive et al. 2005, 2009) and materialization of ontological views (Bhatt et al. 2006) etc. Using ontologies, the integration of information distributed among different applications gives the information integration a new level of automation and flexibility ultimately leading to better description, explanation, conjunction, integration and reasoning on some related data, thereby leading to better decision making.

Currently, the use of ontologies for semantic information integration can be viewed from two perspectives. Firstly, ontologies were introduced as a shared, explicit specification of a conceptualization of a domain. Therefore, ontologies lead to integration tasks to describe the semantics of information sources and to make the content explicit. This also focuses on the design and development of common ontologies that can be extended for more specific application domain specification. However, this will exacerbate the integration problem (Noy 2004; Benkö et al. 2003; Buccella et al. 2009; Xue et al. 2009). Secondly, ontologies with extended rules are used for reasoning purposes. This involves an extension of ontologies with rules, where inference and reasoning are central to the process. Here, rules are defined on top of ontologies to infer new knowledge. The proposals for integration of rules languages and ontology languages can be classified by the degree of integration (Antoniou et al. 2005). Firstly, the hybrid approach is one where there is strict separation between the rule predicates and ontology predicates and reasoning is done by interfacing the existing rule reasoner with the ontology reasoned; whereas, with the homogeneous approach, both rules and ontologies are embedded in the same logical language $\mathcal{L}$ without making a prior distinction between the rule predicates and ontology predicates, and the reasoning single reasoner can be used for reasoning purposes.

Many DSS applications are built using the second approach, i.e. ontology with extended rules, to ensure the availability of integrated, high quality information for decision support. Broadly speaking, such approaches fall into two categories:

1. Semantic Web-DSS
2. Defeasible logic-based Web-DSS

In the following sections, we discuss these categories in detail.

2.1 Semantic Web-DSS

The importance of Semantic-based Web DSS in business applications has been identified by a number of researchers over a period of time (Vahidov and Kersten 2004; Silverman et al. 2001; Toni 2007). Kartha and Novstrup (2009) have proposed a combination of ontologies and decision rules for building a decision support application for time sensitive targeting. They have represented knowledge with the help of rules known as decision rules which: (a) include primitives from multiple ontologies and primitives that are defined by algorithms that are outside of the rule framework; (b) are time-dependent; and (c) incorporate default assumptions. They have developed what is known as the Sentinel system, which is general enough to support a wide variety of DSS tasks.

Ceccaroni et al. (2004), present an environmental decision support system (called OntoWEDSS) for waste water treatment to improve the diagnosis of faults in a treatment plant, which provides support for complex problem-solving and facilitates knowledge modelling and reuse. The system is based on the integration of case-based and rule-based reasoning with an ontology, i.e. Waste-Water Ontology (WaWO), for the representation of the domain and for reasoning. Nicolicin-Georgescu et al. (2010) present an approach to managing data warehouse cache allocations via decision support systems, by using autonomic computing and semantic web technologies. He has presented heuristics for autonomic computing adoption, using ontologies for DSS system modelling and ontology-based rules for heuristics implementation.

Similarly, Salam (2007) presents a supplier performance contract monitoring and execution DSS, using OWL-DL[1] for knowledge representation SWRL[2] to express rules on top of OWL-DL ontologies. Cheung and Cheong (2007) address the challenges of market operations using a rule-based approach in mission-critical decisions and Garcia-Crespo et al. (2011) propose a semantic model for knowledge representation in e-business. Yang et al. (2009) have proposed a Semantic Web-DSS and provide semantics for defining static and dynamic semantics representation based on ontology and quantitative decision making comprising three steps: publishing decision requirement, bidding and then role-based collaboration among decision peers (each Semantic Web-DSS is a peer) to negotiate for decision models.

In all these attempts, the systems integrate information through reasoning with the help of ontologies under certain assumptions including:

1. The given problem can be fully addressed with available information (solution to the problem lies within the available information). In order to elucidate it, let us consider an example. A department in an enterprise wants to improve its product and would like to make use of all the information it holds internally in order to adequately identify issues regarding product quality and improve the product's quality. The department ignores any information outside its own boundaries.
2. The information or specification of business rules for decision-making is consistent. In other words, it is assumed that no contradictory rules will emerge during the decision-making process
3. New information will be consistent with the already available information or specifications.
4. New information does not lead to retraction of previous conclusions.

In the existing literature, there is no research on enterprise-wide Web-DSS that addresses the aforementioned issues of information integration for intelligent decision-making.

2.2 Defeasible logic-based Semantic Web-DSS

This is the second category of Web-DSS having the capability of integrating information which could be incomplete and inconsistent. In this type of Web-DSS, the special types of rules known as defeasible rules are deployed to incorporate defeasible or non-monotonic behaviour in the system.

Dr Prolog (Antoniou and Bikakis 2007) is a Prolog-based implementation for carrying out defeasible reasoning on the semantic web. It provides declarative system support rules, facts, ontologies, RuleML, and both monotonic and non-monotonic rules. The system provides a number of variants such as ambiguity blocking, ambiguity propagation and conflicting literals.

Dr-Device (Kontopoulos et al. 2011; Bassiliades et al. 2004) is CLISP-based defeasible reasoning implementation for information integration provided with a VDR-Device reasoning system. Compared to Prolog, Dr-Device supports only one variant for information integration, i.e. ambiguity blocking.

Sweetjess (Grosof et al. 2002) is another defeasible reasoning system based on Jess and closely resembles courteous logic programs. It allows for procedural attachment and it implements only one reasoning variant. Moreover, it imposes a number of restrictions on the programs so that it can map on Jess.

Table 1 compares different defeasible logic-based semantic web implementations. In the context of semantic Web-DSS, these implementations have various limitations. Firstly, they provide either data-driven or goal-driven reasoning. Data-driven moves from current facts to a certain conclusion, whereas goal-driven reasoning is used to validate the conclusion with supporting facts and answer the user queries. However, in the case of semantic Web-DSS, both types of reasoning are needed for information integration. The existing proposed approaches cannot handle both types of reasoning for information integration. Secondly, they define

[1] http://www.w3.org/TR/owl-guide/

[2] http://www.w3.org/Submission/SWRL/

Table 1 Comparison of defeasible logic based semantic Web-DSS

	Dr-Prolog	Dr-Device	Situated courteous logic
Language	Prolog	JESS	JESS
Logic	Defeasible logic	Defeasible logic	Situated courteous logic
Semantic data	RDFS/OWL	RDF	DAML+OIL
RuleML	Yes	Yes	Yes
Incomplete knowledge representation	Yes	Yes	Yes
Conflict representation	Yes	Yes	Yes
Data-driven reasoning	No	Yes	Yes
Goal-driven reasoning	Yes	No	No
Conflict resolution	User defined individual preferences	User defined individual preferences	User defined individual preferences
Explanation	Textual	Textual	Textual
AIF reification	No	No	No
Information integration	Limited	Limited	Limited
Knowledge integration	No	No	No

explicit (user- defined) individual preferences among conflicting rules at compile time to resolve conflicts between them. The use of these priorities is usually embedded in the derivation mechanism and conflicting rules are compared individually during the derivation process. In such formalisms, the derivation notion is bound to one single comparison criterion. However, the semantic Web-DSS is a source of defeasible knowledge as it is open by nature and subject to inconsistencies deriving from multiple sources; therefore, it is not possible to define priorities in advance among conflicting rules and even if priorities exist, it is not appropriate to compare rules individually during the derivation process. As a result, these systems provide limited information integration and no knowledge integration at all. Additionally, all of them provide to the end user only a textual explanation about the integrated information, and the integrated information results are not exportable in Argument Interchange Format (AIF).

3 Case study

Let us assume that ABC is an enterprise comprised of different departments such as IT, Marketing, and Human Resources. Due to certain unavoidable circumstances, the enterprise has decided to relocate its departments to a new site. Higher authorities have instructed the managers of each department to give their recommendations along with justifications about the suitability of using the XYZ relocation service provider for relocation purposes. The XYZ relocation service provider has its business-related information published on the web, giving all the necessary information required potential customer such as ABC enterprise. Figure 2 depicts the interaction between the internal environment of enterprise ABC and the external environment.

In order to generate recommendations for XYZ, the managers need an automated information integration system which could automatically access, reason and integrate XYZ business information along with their departmental information (requirements). It is important to note here that the information from service provider XYZ could be potentially incomplete and contradict departmental information of enterprise ABC. Additionally, the managers also want to incorporate user feedback provided by existing XYZ clients via a feedback forum, in the decision-making process. On successful information integration, managers need to forward their recommendations to higher authorities in a standard format.

Once each department within the ABC enterprise forwards its recommendation to higher authorities, the

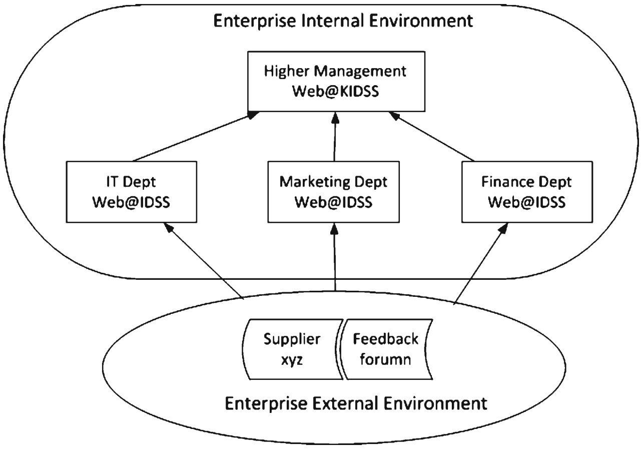

Enterprise Internal Environment
Higher Management Web@KIDSS
IT Dept Web@IDSS
Marketing Dept Web@IDSS
Finance Dept Web@IDSS
Supplier xyz
Feedback forumn
Enterprise External Environment

Fig. 2 Interaction of enterprise ABC with external environment

latter will need to integrate the knowledge or recommendations about XYZ obtained from each department into a coherent information model which could help them to reach a final decision, i.e. whether or not to engage the services of the XYZ relocation service provider.

Here we formalize the requirements for enterprise ABC to successfully achieve the aforementioned tasks.

- A declarative language for specifying the business requirements of an organization
- Language with the capability of representing incomplete and conflicting information (i.e. business rules and data)
- Information integration via an inference mechanism that can perform reasoning pertaining to incomplete and conflicting information coming from different sources.
- Justifiable explanation of the information integration results
- Ability to export results to other software systems
- Knowledge integration through a user-defined scheme that allows the user to define constraints pertaining to knowledge to be integrated.

Assumption

- Organizations ABC, XYZ and Feedback forum share a common vocabulary defined in OWL/RDF format and the predicates defined in the vocabulary are used for the specification of business rules.

4 Argumentation and argumentation schemes

In everyday life communication, argumentation often has negative connotations, suggesting quarrelsomeness and unpleasantness; this is a misconception. In fact, argumentation in the classical sense is the study of effective reasoning which is a key to the way that humans deal with conflicting information by taking into account arguments and counter-arguments relevant to a certain issue (Zarefsky 2009). Argumentation is inherently a process rather than an instant picture, and the building blocks of argumentation are the arguments and the relationship between those arguments (Loui 1998). According to Walton (2009) and Palau and Moens (2009), an argument is a set of statements (propositions) consisting of a conclusion, a set of premises, and inference from premises to conclusion. During the process of argumentation, relationships among the arguments are linked with each other in a certain pattern to support the ultimate conclusion. Such linking patterns are called ‘argumentation schemes’ and allow reasoning to be performed using a set of premises and a conclusion. These argumentation schemes have emerged from informal logic (Godden and Walton 2007). The schemes help to categorize the way that arguments are built. They bridge the gap between logic-based application and human reasoning by capturing stereotypical patterns of human reasoning. An example is an argument from an expert opinion scheme. Formally, an argumentation scheme is composed of a set of premises A_i, a conclusion denoted as S, and a set of critical questions CQ_i aimed at defeating the derivation of the consequent (Letia and Groza 2008; Rahwan et al. 2007a).

Argumentation formalism has been used in the past by different researchers in DSS for practical reasoning. Morge (2008) proposed a DSS based on abductive reasoning which helps the decision maker to select a business location after evaluating different alternatives; it suggests several solutions and provides an interactive and intelligible explanation of those choices. To develop the DSS, Morge (2008) used a logic language to represent knowledge, goals and actions with quantitative priorities attached to them to represent likelihood of knowledge, preferences between goals and expected utilities of actions respectively. Similarly, Chesnevar et al. (2006b) identified that the current critic recommender systems are incapable of dealing with the defeasible nature of information. They present a novel approach to the integration of DSS, such as critics and recommender systems with a defeasible argumentation framework, to enhance the practical reasoning capabilities of such systems.

5 Argumentation enabled semantic information integration

The use of ontologies have helped to improve the integration of information derived from different sources to a new level of automation and flexibility. This has ultimately led to better description, explanation, conjunction, integration and reasoning on some related data, resulting in better decision-making. However, the current generation of Web-DSS is not able to represent, reason and integrate incomplete and inconsistent information for information integration purposes, irrespective of whether this information emerges from within the organization or outside it (Carlsson and Turban 2002).

We have addressed this challenge (representation and reasoning over incomplete and inconsistent semantic information) by using Web@IDSS (Janjua and

Hussain 2011). The Web@IDSS uses Defeasible logic-based argumentation formalism (DeLP)(Garcia and Simari 2004) as knowledge representation and reasoning language with certain extensions.

$$\mathcal{WM} = \left\{ \begin{array}{c} relocationService(xyz), client(it), useService(xyz), efficient(xyz), safeDelivery(xyz) \\ language(english), languageProblem(xyz, english) \end{array} \right\} illustration(1)$$

$$\mathcal{R} = \left\{ \begin{array}{l} [x.d1]relocationService(X), client(Y), useService(X) \dashrightarrow \sim giveDiscount(X) \\ [x.d2]client(Y), relocationService(X), reuseService(Y, X) \dashrightarrow giveDiscount(Y) \\ [x.d3]efficient(X), safeDelivery(X) \dashrightarrow reliableService(X) \\ [x.s1]giveDicount(Y), advancmentPayment(Y) \rightarrow normalDiscount(Y) \\ [x.s2]normalDiscount(Y), bulkOrder(Y) \rightarrow platinumDiscount(Y) \\ [a.it.d1]client(Y), happy(Y, X), relocationService(X) \dashrightarrow resuseService(Y, X) \\ [a.it.d3]ontimeDelivery(X) \dashrightarrow efficient(X). \\ [rc1.a.it.d4]not\ dmanageProduct(X) \dashrightarrow safeDelivery(X) \\ [a.it.d5]largeTruck(X), reuseService(Y, X), reliableService(X), normalDiscount(Y) \\ \dashrightarrow goodRelocationService(X) \\ [a.it.d6]language(ENG), languageProblem(X, ENG) \dashrightarrow \sim clearCriteria(X) \\ [a.it.d7]demandCash(X), demandTip(X) \dashrightarrow \sim convienent(xyz) \\ [a.it.d8]goodRelocationService(X), not\ convienent(X), not\ clearCriteria(X) \\ \dashrightarrow recommendService(X) \end{array} \right\} illustration(2)$$

$$Args = \{\} \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots illustration(3)$$

The system is capable of translating business rules defined in RuleML[3] syntax into DeLP rules. The system also translates the RDF/XML data into DeLP facts and RDF(S) and part of OWL Ontologies into DeLP facts and DeLP rules. Once the information such as business rules and data translation is complete, the system starts an argumentation construction cycle.

Lets us assume that illustrations 1, 2 and 3 depicts the initial state of the argumentative production system of the IT department of enterprise ABC. Illustration 1 represents current information saved in the working memory ($\mathcal{WM}$) and illustration 2 represents business policies i.e. business rules ($\mathcal{R}$) of the IT department and XYZ service provider, each identified by their respective label, and illustration 3 depicts an empty active argument set ($Args$) before the arguments construction cycle.

$$\mathcal{WM}' = \left\{ \begin{array}{c} relocationService(xyz), client(it), useService(xyz), efficient(xyz), safeDelivery(xyz) \\ language(english), languageProblem(xyz, english), resuseService(it, xyz), giveDiscount(it), \\ normalDiscount(it), reliableService(xyz), goodRelocationService(xyz), \sim clearCriteria(xyz), \\ \sim convienent(xyz), recommendService(xyz) \end{array} \right\} illustration(4)$$

$$Args = \left\{ \begin{array}{l} [rc1.a.it.d1]client(it), happy(it, xyz), relocationService(xyz) \dashrightarrow reuseService(it, xyz) \\ [rc1.x.d2]client(it), relocationService(xyz), reuseService(it, xyz) \dashrightarrow giveDiscount(it); \\ [rc1.x.s1]giveDicount(it), advancmentPayment(it) \rightarrow normalDiscount(it) \\ [rc1.a.it.d3]ontimeDelivery(xyz) \dashrightarrow efficient(xyz). \\ [rc1.a.it.d4]not\ dmanageProduct(xyz) \dashrightarrow safeDelivery(xyz). \\ [rc1.x.d3]efficient(xyz), safeDelivery(xyz) \dashrightarrow reliableService(xyz) \\ [rc1.a.it.d5]largeTruck(xyz), reuseService(it, xyz), reliableService(xyz) \\ , normalDiscount(xyz) \dashrightarrow goodRelocationService(xyz) \\ [rc1.a.it.d6]language(english), languageProblem(xyz, English) \dashrightarrow \\ \sim clearCriteria(xyz) \\ [rc1.a.it.d7]demandCash(xyz), demandTip(xyz) \dashrightarrow \sim convienent(xyz) \\ [rc1.a.it.d8]goodRelocationService(xyz), not\ convienent(xyz), \\ not\ clearCriteria(xyz) \dashrightarrow recommendService(xyz) \end{array} \right\} illustration(5)$$

[3] http://ruleml.org

The argument construction is a recursive process which involves interpretation of business rules. The system searches for business rules from a knowledge base whose pattern matches the facts present in working memory ($\mathcal{WM}$) and, on a successful match, executes the business rule which then adds the rules conclusion, i.e. ground predicate, to the working memory. The argument construction process continues until all the matched business rules in the knowledge base have been processed. This interpretation of a business rule is also known as 'firing of a rule'. Illustrations 4 and 5 depicts the updated argumentative production system of the IT department with a populated active argument set and updated working memory($\mathcal{WM}\prime$).

The argument construction followed by the argumentation phase consists of conflict detection, resolution and justified explanation. The conflict between an argument and its counter-argument is resolved either by static static priority establishment or dynamic priority establishment. For dynamic priority establishment, the process of argumentation starts where an argument may be defeated by other arguments. Since counter-arguments are also arguments which in turn may be defeated, this process results in the construction of dialectical trees. This is an interesting property of the argumentation approach which involves dialectical proof procedures that are quite close to the process used by humans when discussing an issue. This similarity to human-style discussions gives argumentation an advantage that can be useful in many contexts. As a result of argumentation, the development of dialectical analysis for conflict resolution amongst arguments leads to the establishment of dynamic priorities between conflicting arguments.

The last step in the argumentation process is the construction of reasoning chains. During this process, all the sub-arguments of an argument including arguments in conflict with undefeated dialectical trees are linked together as a reasoning chain. This process continues until all possible arguments are linked up into reasoning chains. The top argument i.e. conclusion, of the reasoning chain is known as 'the 'result' of the reasoning chain', and the chain of sub-arguments supporting the top argument is called the 'support' of the result. Figure 3 is a graphical representation of a reasoning chain of the argumentative production system depicted in illustrations 4 and 5. The arguments in Fig. 3 are represented in short form e.g. $[rc1.a.it.d8]recommendService(xyz)$ where [rc1.a.it.d8] is the label of the argument and $recommendService(xyz)$ is the claim of the argument.

5.1 Argument interchange format (AIF) reification

The AIF is an international effort to develop a representational mechanism for exchanging argument resources between research groups, tools, and domains using a semantically rich language (Chesnevar et al. 2006a; Iyad Rahwan 2009; Rahwan et al. 2007b). The AIF was developed as a commonly agreed upon core ontology i.e. AIF ontology that specifies the basic concepts used to express arguments and the relationship between arguments. Figure 4 depicts the AIF core ontology. The upper ontology describes the arguments in the form of a connected network of nodes. The nodes are of two types, namely, information nodes (I-Node) and scheme nodes (S-nodes). The I-Node captures the information in the form of premise, conclusion, exception or presumption. The scheme nodes

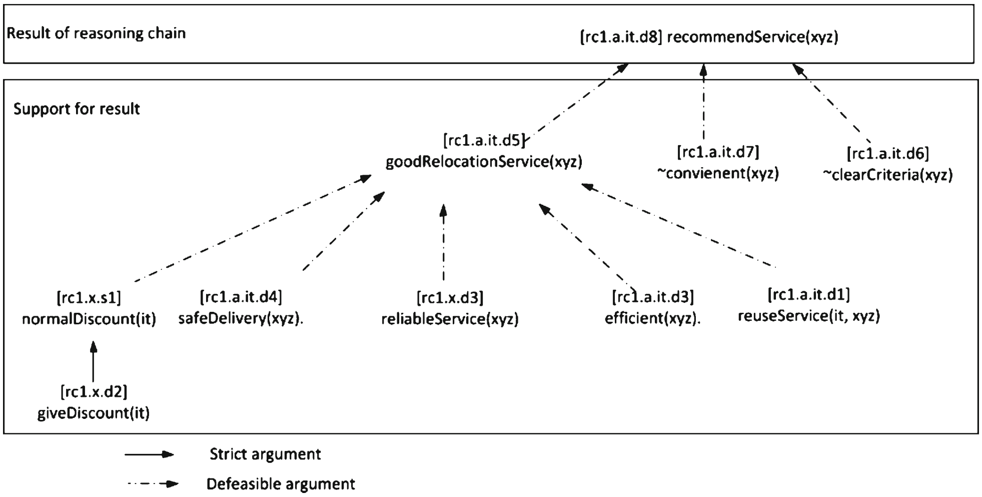


Fig. 3 Graphical representation of a reasoning chain

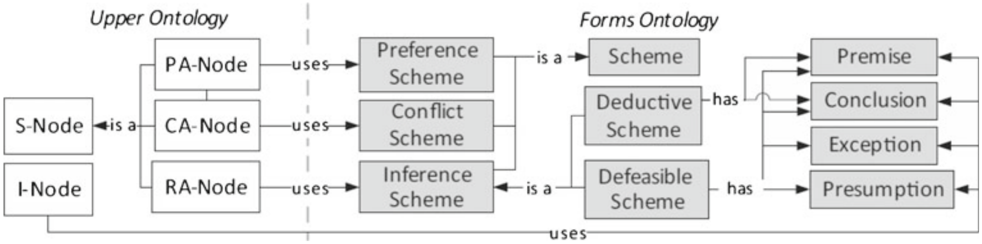


Fig. 4 The upper and forms ontologies of the AIF (Bex et al. 2010)

provide the relationship between two I-Nodes and are further classified as rule application nodes (RA-Node) that correspond to inference from premises to claim, conflict nodes (CA-Node) that correspond to conflict between two nodes and preference application nodes (PA-node) that correspond to preference ordering between conflicting nodes. These different kinds of nodes are used to build the AIF argument graph defined by Rahwan et al. (2007b) as follows:

Definition 1 (Argument Network) An argument network Φ is a graph G consisting of

- a set $\mathcal{N}$ of vertices (or nodes) comprises of I-Nodes and S-Nodes; and
- a binary relation $\xrightarrow{\text{edge}}$: $\mathcal{N}$ x $\mathcal{N}$ representing edges among nodes

such that $\nexists\, (i, j) \in \xrightarrow{\text{edge}}$ where both $i \in \mathcal{N}_1$ and $j \in \mathcal{N}_1$.

The I-Node can be connected to other I-Nodes only through S-Nodes to represent a rationale behind the relationship between I-Nodes. However, the S-Node can be directly connected to other S-Nodes to represent some kind of meta-reasoning. For instance, RA-to-RA and RA-to-PA edges might indicate some kind of meta-justification for the application of an inference rule or a particular criterion for defining preference. Additionally, the ontology does not use typed edges in a graph; instead, the semantics for edges can be inferred node types that they connect. The argumentation schemes are represented as forms of arguments known as 'forms ontology' in AIF core ontology as depicted in Fig. 4. Those argumentation schemes are preference scheme, conflict scheme and inference scheme. The inference scheme is again divided into a deductive scheme which represents deductive inference from premises to a claim, and a defeasible scheme which represents defeasible inference from premises to a claim.

The AIF reification in the Web@IDSS will help to express arguments in more concrete language, and with better representation and handling of conflicts in an argument network and better evaluation of complex arguments and reasoning chains. The export of results provided by Web@IDSS in AIF-compliant format will enable the system to merge it with the justified conclusions of other machines in the absence of complete or accurate information.

In the rest of this section, we provide formalisms that make use of an 'Argument Network' to represent integrated information as argumentative reasoning chains.

Definition 2 (Web@IDSS argument network) Given an argument graph G and set of forms F in a argument network Φ, a Web@IDSS argument network AG is defined as follows: $(\mathcal{WM}, \mathcal{R}, Args)$ Where

- $\mathcal{WM}$: a set of information nodes i.e. $\mathcal{N}^I_{i,\ldots,n}$, where I represents information node and i represents index of the node.
- $\mathcal{R}$: a set of user-defined rules or specifications to establish links between $\mathcal{N}^I_i$ nodes through S node such that $\nexists\, (i, j) \in \xrightarrow{\text{edge}}$ where both $i \in \mathcal{N}_1$ and $j \in \mathcal{N}_1$
- $Args$ a set of arguments derived from $\mathcal{R}$, where each argument establishes a linked set of premise $(\mathcal{N}^I_i)$ to a claim$(\mathcal{N}^I_j)$ through S node. Based upon the forms of ontology, the strict argument and defeasible argument are defined as follows:

(Strict argument) :

$$\mathcal{N}^I_i, \ldots\ldots, \mathcal{N}^I_j \overset{\text{Uses(RA,deductiveScheme)}}{\longrightarrow} \mathcal{N}^I_k$$

(Defeasible argument) :

$$\mathcal{N}^I_m, \ldots\ldots, \mathcal{N}^I_n \overset{\text{Uses(RA,defeasibleScheme)}}{\dashrightarrow} \mathcal{N}_o$$

The binary relation $\xrightarrow{\text{edge}}$: $\mathcal{N}$ x $\mathcal{N}$representing edge among nodes in Web@IDSS can be categorized as follows:

- Counter-argument: $\mathcal{N}^I_i \overset{\text{Uses(CA-Node)}}{\dashrightarrow} \sim \mathcal{N}^I_j$ such that $\mathcal{N}^I_i$ is counter-argue $\mathcal{N}^I_j$

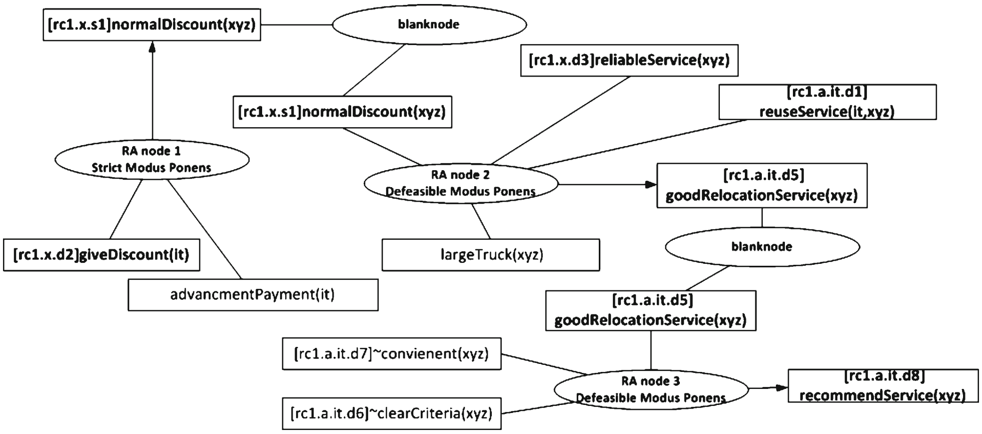

[rc1.x.s1]normalDiscount(xyz)
blanknode
[rc1.x.d3]reliableService(xyz)
[rc1.a.it.d1] reuseService(it,xyz)
[rc1.x.s1]normalDiscount(xyz)
RA node 1 Strict Modus Ponens
RA node 2 Defeasible Modus Ponens
[rc1.a.it.d5] goodRelocationService(xyz)
[rc1.x.d2]giveDiscount(it)
largeTruck(xyz)
blanknode
advancmentPayment(it)
[rc1.a.it.d5] goodRelocationService(xyz)
[rc1.a.it.d7]~convienent(xyz)
RA node 3 Defeasible Modus Ponens
[rc1.a.it.d8] recommendService(xyz)
[rc1.a.it.d6]~clearCriteria(xyz)

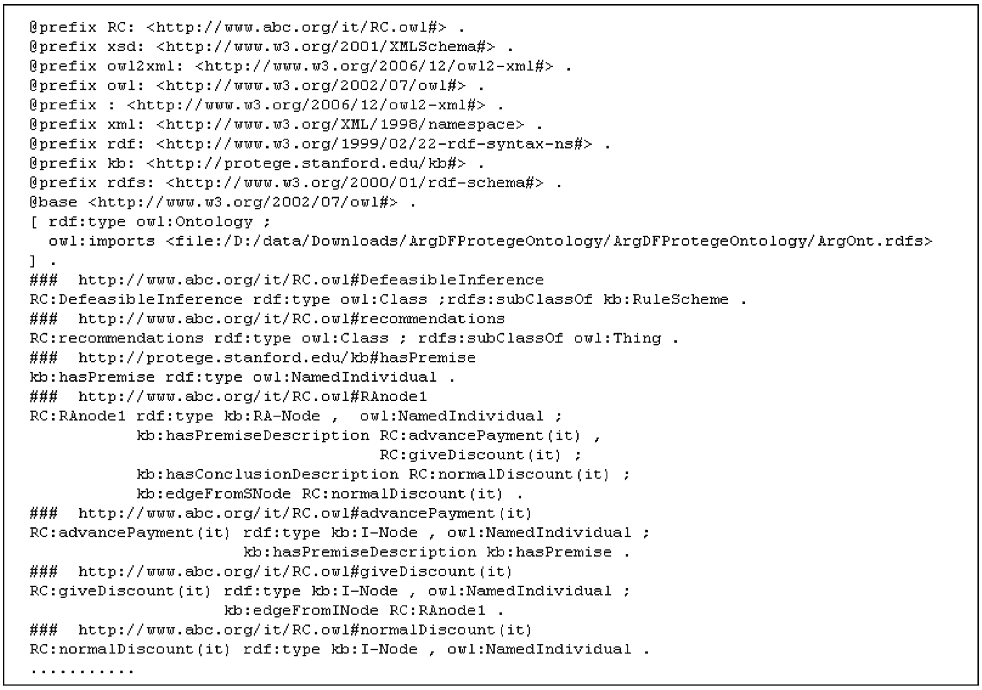
Fig. 5 Pictorial representation of a AIF compliant reasoning chain

- Static defeat: $\mathcal{N}_i^I \overset{\text{Uses(PA−Node)}}{\dashrightarrow} \mathcal{N}_j^I$ such that $\mathcal{N}_i^I$ is has priority over $\mathcal{N}_j^I$
- Dynamic defeat: $\mathcal{N}_i^I \overset{\text{Uses(PA−Node)}}{\dashrightarrow} \mathcal{N}_j^I$ such that $\mathcal{N}_i^I$ has priority over $\mathcal{N}_j^I$
- Sub-argument: For representation of sub-argument relationship in AIF format, we added a blank-node into argument network i.e. $\mathcal{N}_i^I \overset{\text{Uses(Blank−Node)}}{\dashrightarrow} \mathcal{N}_j^I$ such that $\mathcal{N}_i^I$ (claim of an argument) is sub-argument of $\mathcal{N}_j^I$ (premise of an argument).

Definition 3 (Predecessor and Successor Nodes) Given a graph AG consisting of a set of nodes $\mathcal{N}$ and a relation $\mathcal{S} \subseteq \mathcal{N} \times \mathcal{N}$ defining the set of edges between the nodes. For each node n $\in \mathcal{N}$, we define the set of its predecessor and successor nodes as follows:

- A Predecessor node: $\{x \in \mathcal{N} \mid (x, n) \in \mathcal{S}\}$,
- A Successor node: $\{x \in \mathcal{N} \mid (n, x) \in \mathcal{S}\}$.

Using the above definitions, we represent the reasoning chains produced by Web@IDSS in AIF format. The node $\mathcal{N}_i^I$ with no successor and have predecessor nodes



```
@prefix RC: <http://www.abc.org/it/RC.owl#> .
@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .
@prefix owl2xml: <http://www.w3.org/2006/12/owl2-xml#> .
@prefix owl: <http://www.w3.org/2002/07/owl#> .
@prefix : <http://www.w3.org/2006/12/owl2-xml#> .
@prefix xml: <http://www.w3.org/XML/1998/namespace> .
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix kb: <http://protege.stanford.edu/kb#> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
@base <http://www.w3.org/2002/07/owl#> .
[ rdf:type owl:Ontology ;
  owl:imports <file:/D:/data/Downloads/ArgDFProtegeOntology/ArgDFProtegeOntology/ArgOnt.rdfs>
] .
###  http://www.abc.org/it/RC.owl#DefeasibleInference
RC:DefeasibleInference rdf:type owl:Class ;rdfs:subClassOf kb:RuleScheme .
###  http://www.abc.org/it/RC.owl#recommendations
RC:recommendations rdf:type owl:Class ; rdfs:subClassOf owl:Thing .
###  http://protege.stanford.edu/kb#hasPremise
kb:hasPremise rdf:type owl:NamedIndividual .
###  http://www.abc.org/it/RC.owl#RAnode1
RC:RAnode1 rdf:type kb:RA-Node ,  owl:NamedIndividual ;
           kb:hasPremiseDescription RC:advancePayment(it) ,
                                    RC:giveDiscount(it) ;
           kb:hasConclusionDescription RC:normalDiscount(it) ;
           kb:edgeFromSNode RC:normalDiscount(it) .
###  http://www.abc.org/it/RC.owl#advancePayment(it)
RC:advancePayment(it) rdf:type kb:I-Node , owl:NamedIndividual ;
                      kb:hasPremiseDescription kb:hasPremise .
###  http://www.abc.org/it/RC.owl#giveDiscount(it)
RC:giveDiscount(it) rdf:type kb:I-Node , owl:NamedIndividual ;
                    kb:edgeFromINode RC:RAnode1 .
###  http://www.abc.org/it/RC.owl#normalDiscount(it)
RC:normalDiscount(it) rdf:type kb:I-Node , owl:NamedIndividual .
...........
```

Fig. 6 Serialization of AIF compliant reasoning chain in turtle format

is called the ‘result’ of reasoning chain. The remaining nodes are known as ‘support’ for the result. Figure 5 depicts the graphical representation of a reasoning chain in AIF format.[4]

5.2 Publication of reasoning chains

The purpose of AIF reification is to publish and share the results of a Web-DSS over the web, enterprise intranet or with other applications in order to provide better decision-making support. For annotation of a reasoning chain, we developed a ‘reasoning chain ontology’ on top of the ArgDF ontology[5] and serialized the AIF-compliant reasoning chain in RDF/XML format. Figure 6 depicts the serialization of a reasoning chain in turtle format.

6 Argumentation-scheme-enabled, argumentation-driven semantic knowledge integration

Today, the decision-making environment has become very complex and decentralized, exacerbated by WWW. Results produced by one Web-DSS might need to be integrated with other Web-DSS to obtain a comprehensive picture of the problem at enterprise level to enable higher authorities to gain business insights and make better decisions. We call such information integration that is related to one subject and distributed among different information sources ‘knowledge integration’. Let us consider the case study discussed in Section 3 where each department needs to formulate and forward its recommendations about the relocation service provider XYZ to higher authorities. During this process, each department, with the help of Web@IDSS or AIF-compliant DSS system, produces recommendations in the form of a reasoning chain. Let us assume that according to the IT department, although the relocation service provider is not convenient and not good at formalising the clients’ criteria, still we assume it is a good relocation service provider and we recommend it. Whereas, other departments have a different opinion. According to illustrations 6, 7 and 8 which depict the recommendations produced by the IT, Marketing and Human Resources departments respectively, it is quite evident that each department has some valuable information about relocation service supplier XYZ, which could help the higher authorities to make the final decision about this supplier. But the biggest challenge is how to automate the integration of this knowledge which is derived from different sources and could be incomplete and inconsistent, to facilitate the decision making process at enterprise level.

$$IT = \left\{ \begin{array}{l} [rc1.a.it.d1] client(it), happy(it, xyz), relocationService(xyz) \dashrightarrow reuseService(it, xyz) \\ [rc1.x.d2] client(it), relocationService(xyz), reuseService(it, xyz) \dashrightarrow giveDiscount(it); \\ [rc1.x.s1] giveDicount(it), advancmentPayment(it) \rightarrow normalDiscount(it) \\ [rc1.a.it.d3] ontimeDelivery(xyz) \dashrightarrow efficient(xyz). \\ [rc1.a.it.d4] not\ dmanageProduct(xyz) \dashrightarrow safeDelivery(xyz). \\ [rc1.x.d3] efficient(xyz), safeDelivery(xyz) \dashrightarrow reliableService(xyz) \\ [rc1.a.it.d5] largeTruck(xyz), reuseService(it, xyz), reliableService(xyz), \\ normalDiscount(xyz) \dashrightarrow goodRelocationService(xyz) \\ [rc1.a.it.d6] language(english), languageProblem(xyz, english) \dashrightarrow\sim clearCriteria(xyz) \\ [rc1.a.it.d7] demandCash(xyz), demandTip(xyz) \dashrightarrow\sim convienent(xyz) \\ [rc1.a.it.d8] goodRelocationService(xyz), not\ convienent(xyz), not\ clearCriteria(xyz) \\ \dashrightarrow recommendService(xyz) \end{array} \right\} illustration(6)$$

[4] The directed arrow are just to emulate the edge from S-node to N-node claim of the argument.

[5] http://www.argdf.org/source/ArgDFProtegeOntology.zip

$$Mar = \left\{ \begin{array}{l} [rc2.a.mk.d1] not\ happy(marketing, xyz), relocationService(xyz) \dashrightarrow \sim reuseService(marketing, xyz) \\ [rc2.x.d1] relocationService(xyz), client(marketing), useService(xyz) \dashrightarrow \\ \sim giveDiscount(marketing) \\ [rc2.a.mk.d3] ontimeDelivery(xyz), largeTruck(xyz), \dashrightarrow efficient(xyz) \\ [rc2.a.mk.d5] dmanageProduct(xyz) \dashrightarrow \sim safeDeliver(xyz) \\ [rc2.a.mk.d4] not\ efficent(xyz), not\ reuseService(xyz), not\ giveDiscount(marketing), \\ not\ safeDeliver(xyz) \dashrightarrow \sim goodRelocationService(xyz) \\ [rc2.a.mk.d6] not\ goodRelocationService(xyz) \dashrightarrow \sim recommendService(xyz) \end{array} \right\} illustration(7)$$

$$HR = \left\{ \begin{array}{l} [rc3.a.hr.d1] language(english), languageProblem(xyz, english) \dashrightarrow \sim clearCriteria(xyz) \\ [rc3.x.d2] client(hr), relocationService(xyz), reuseService(hr, xyz) \dashrightarrow giveDiscount(hr) \\ [rc3.a.hr.d2] not\ ontimeDelivery(xyz) \dashrightarrow \sim efficient(xyz) \\ [rc3.a.hr.d3] not\ efficient(xyz), not\ giveDiscount(xyz) \dashrightarrow \sim goodRelocationService(xyz). \\ [rc3.a.hr.d4] not\ goodRelocationService(xyz), not\ clearCriteria(xyz), \dashrightarrow \\ \sim recommendService(xyz) \end{array} \right\} illustration(8)$$

In the following section, we define the formal syntax and semantics for Argumentation scheme enabled Argumentative Knowledge Integration Web-DSS (Web@KIDSS). The proposed system is capable of integrating knowledge regarding one subject derived from different Web-based DSS into a coherent, consolidated form so that higher authorities can have better business insights and make the optimal decisions or decisions that take a broader perspective.

6.1 Formal syntax and semantics

Definition 4 (Recommendation space) A collection of recommendations, each in the form of a reasoning chain $\lambda_{(\mathrm{identifer,result})}$ contributed by a source '*i*' is known as a 'recommendation space'. Mathematically, recommendation space is defined as follows:

$$\Theta = \sum_{i=0}^{n} \left\{ [i] \lambda_{(\mathrm{identifer,result})} \right\} \tag{1}$$

The recommendation space' for enterprise ABC is depicted in Fig. 7 can be mathematically represented as follows:

$$\Theta = \left\{ [rc1]\lambda_{(\mathrm{A,recommend})}, [rc2]\lambda_{(\mathrm{B,\sim recommend})}, [rc3]\lambda_{(\mathrm{C,recommend})} \right\} \tag{2}$$

Where

- $[rc1]\lambda_{(A,a)}$ represents the recommendation in the form of a reasoning chain by department IT department identified as rc1.
- Similarly, $[rc2]\lambda_{(B,\sim a)}$ is a recommendation from the HR department identified as rc2 and $[rc3]\lambda_{(C,a)}$ is a recommendation from the marketing department identified as rc3.

Definition 5 (Integration scheme) An Integration scheme, a user-defined argumentation scheme (Katie Atkinson 2008), is a tuple having the following form:

$$\mathcal{IS} = \{\{name, (premise_i, ...premise_n), conclusion, criticalquestions, variant\} \tag{3}$$

Where

- name is the label of the scheme which identifies the scheme
- premise is a set of facts to be matched
- sConclusion is a result of the scheme
- sCriticalquestion is a set of queries
- svaraint is a boolean flag for conflicts blocking. If svariant is true, the conflicts are blocked and the reasoning chain will not considered for any further processing; whereas, if the flag is false, then the reasoning chains with conflicts are still considered for further processing.

The critical questions can be categorized as exceptions and assumptions. The premises provide reasons for accepting the conclusion only if the assumptions are true and there are no exceptions. If either an assumption is false or an exception is true, unless premises provide reasons for accepting the conclusion, the conclusion would not be valid (Katie Atkinson 2008). Thus, both assumption and exceptions attack the conclusion of the scheme.

In the context of the case study discussed above, the integration scheme represents the criteria that are used by higher authorities to evaluate each department's recommendation before their integration. If the recommendation meets the integration criteria, then the reasoning chain is included in the knowledge integration process.

Definition 6 (Valuation operator) The application of the integration scheme to a reasoning chain is termed 'valuation of a reasoning chain'. Mathematically, we define the valuation operator $\between$ as a binary operator such as

$$[rc1]\lambda^{\mathrm{val}}_{(A,a)} = \left\{[rc1]\lambda_{(A,a)} \between \mathcal{IS}\right\} \tag{4}$$

During valuation of a reasoning chain, all the premises and critical questions originating from the integration scheme are executed on the corresponding reasoning chain. If the premises match the reasoning chain and queries return true on execution over the reasoning chain, then the reasoning chain is considered to be a valued reasoning chain. The reasoning chain is still considered valued if the reasoning chain premise does not match or queries return false, but the conflict blocking flag, i.e. svariant is false.

Definition 7 (Focus operator) $\otimes$ is a binary operator, such that

$$[rc1]\lambda^{\mathrm{val}}_{(A,a)} \otimes [rc2]\lambda^{\mathrm{val}}_{(B,a)} \tag{5}$$

is called a 'focus operator'. This corresponds to AND operator. If two arguments, belonging to different reasoning chains, have the same claim, the application of the focus operator produces those arguments in a resultant set. Let us consider the recommendation space depicted in Fig. 7, the application of the focus operator to reasoning chains rc2 and rc3 results in the following set of common claims $\{[mk.d4, hr.d4] \sim recommendService(xyz), [mk.d6, hr.d6] \sim goodRelationService(xyz)\}$.

Definition 8 (Merge operator (Fan et al. 2010)) $\boxplus$ is a binary operator, such that

$$[ar1]a, b, c \dashrightarrow d \boxplus [ar2]\, e, b, c \dashrightarrow d \tag{6}$$

is called a 'merge operator'. This corresponds to the OR operator. Let us consider the recommendation space depicted in Fig. 7, containing two argument 'hr.d6' argument and 'mk.d6' argument belonging to reasoning chain rc2 and rc3 respectively. The application of the merge operator to these arguments results in the construction of a new argument which would look like:

[hr.d6][mk.d6]$\sim goodRelocationService(xyz), \sim clearCriteriac(xyz) \dashrightarrow \sim recommendService(xyz)$.

The merge operator applies to the arguments with the same inference type.

Definition 9 (Unique operator) $\odot$ is a binary operator, such that

$$[rc1]\lambda^{\mathrm{val}}_{(A,a)} \odot [rc2]\lambda^{\mathrm{val}}_{(B,a)} \tag{7}$$

is called a 'unique operator'. The application of unique operator on reasoning chains results in all those arguments whose claim is unique between the reasoning chains. Let us consider the recommendation space depicted in Fig. 7, the application of unique operator on reasoning chains rc2 and rc3 results in following set of arguments $\{\sim clearCriteria(xyz), giveDiscount(mk),$

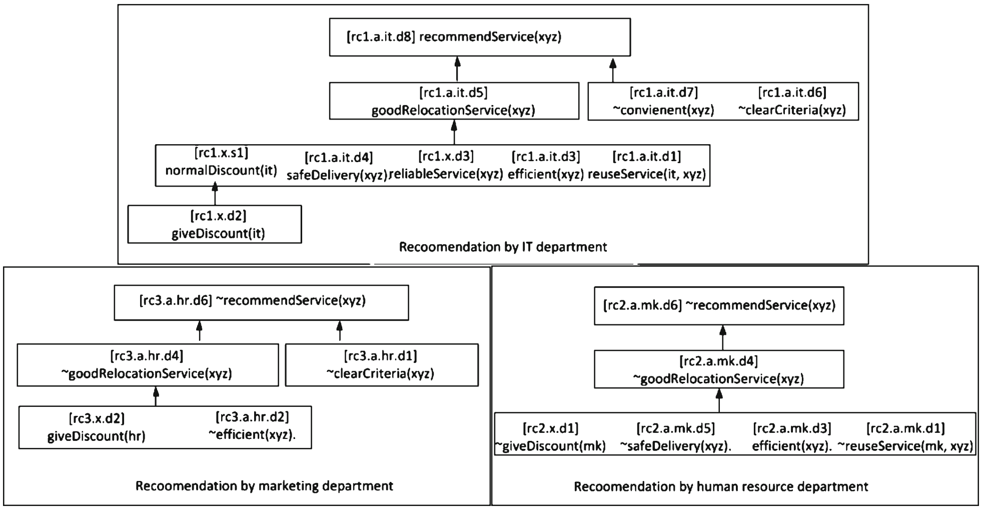

[rc1.a.it.d8] recommendService(xyz)
[rc1.a.it.d5] goodRelocationService(xyz)
[rc1.a.it.d7] ~convienent(xyz)
[rc1.a.it.d6] ~clearCriteria(xyz)
[rc1.x.s1] normalDiscount(it)
[rc1.a.it.d4] safeDelivery(xyz)
[rc1.x.d3] reliableService(xyz)
[rc1.a.it.d3] efficient(xyz)
[rc1.a.it.d1] reuseService(it, xyz)
[rc1.x.d2] giveDiscount(it)
Recoomendation by IT department
[rc3.a.hr.d6] ~recommendService(xyz)
[rc3.a.hr.d4] ~goodRelocationService(xyz)
[rc3.a.hr.d1] ~clearCriteria(xyz)
[rc3.x.d2] giveDiscount(hr)
[rc3.a.hr.d2] ~efficient(xyz).
Recoomendation by marketing department
[rc2.a.mk.d6] ~recommendService(xyz)
[rc2.a.mk.d4] ~goodRelocationService(xyz)
[rc2.x.d1] ~giveDiscount(mk)
[rc2.a.mk.d5] ~safeDelivery(xyz).
[rc2.a.mk.d3] efficient(xyz).
[rc2.a.mk.d1] ~reuseService(mk, xyz)
Recoomendation by human resource department

Fig. 7 Recommendation space

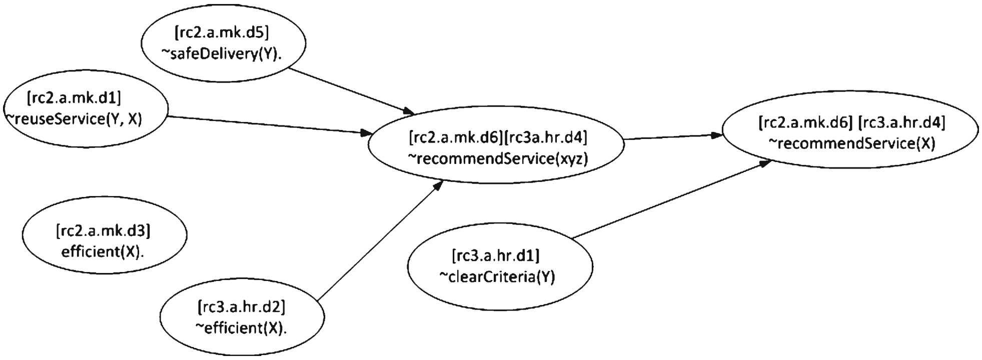


Fig. 8 Graphical representation of integration of rc2 and rc3

$\sim giveDiscount(hr), \sim safeDelivery(xyz), \sim effícent(xyz), \sim reuseService(xyz)\}$.[6]

Definition 10 (Conflict operator) is a binary operator, such that

$$[rc1]\lambda^{val}_{(A,a)} \oslash [rc2]\lambda^{val}_{(B,a)} \tag{8}$$

is called a 'conflict operator'. The application of this operator to reasoning chains will return the set of arguments along with their counter-argument and undefeated or blocking dialectical trees.

Let us consider the recommendation space illustrated in Fig. 7, the application of conflict operator on reasoning chains rc2 and rc3 results in the following set of claims $\{efficient(xyz), \sim efficient(xyz)\}$

Definition 11 (Preference operator) is a binary operator such that

$$[a]giveDiscount(XYZ) > [b] \sim giveDiscount(XYZ) \tag{9}$$

is known as a 'preference operator'. The end user can define a preference relation explicitly for an argument and its counter-arguments.

Definition 12 (Knowledge integration) The result of a reasoning chain $\lambda_{(A,a)}$ supported by a chain of sub-arguments belonging to valued recommendation set produces an Integrated reasoning chain $\Theta_a = (\lambda_i, ..., \lambda_n)$ where $0 < i < n$. Mathematically, we define an integrated reasoning chain as follows:

$$\forall r, s \in valuated\,argument\,set\{if(s\xi r)then\Theta_j = \Theta_j \cup s \tag{10}$$

where ξ is used to represent the sub-argument relationship and $\Theta_j$is used to represent an integrated reasoning chain for result j. Taking into consideration the recommendation space depicted in Fig. 7, the integration of rc2 and rc3 results are shown in Fig. 8 considering conflict blocking flag is false.

7 Proposed conceptual framework

In this section, we elaborate in detail the proposed conceptual framework of Web@KIDSS for knowledge integration spanning across different Web-DSS. The proposed framework takes into consideration the reasoning chains (recommendations) published on the web or enterprise intranet for enabling decision-making based on enterprise-wise information integration. The task of Web@KIDSS is to integrate them into a consolidated reasoning chain depicting an enterprise-wise picture to the decision maker and facilitate the decision-making process. To achieve this task, the system user defines an integration scheme to perform a primary evaluation of a reasoning chain to determine whether it is suitable for integration with the rest of the reasoning chains. The proposed framework uses DeLP as knowledge representation and reasoning language with certain extensions (Janjua and Hussain 2011). Figure 9 depicts the proposed conceptual framework. The key components of our proposed framework are as follows:

7.1 Translation of the reasoning chains

The reasoning chains published on intranet by different Web-DSS in AIF compliant format are imported by the system user and system translates the imported reasoning chains in DeLP compliant reasoning chains. During this process, the translation of nodes and the relationship between nodes in an AIF formatted reasoning chain are translated to DeLP construct. The information nodes are translated as either premise of an argument or claim, whereas scheme nodes are used to build the types of arguments and relationship between arguments. For example, if there is an RA-node (defeasible or strict inference) the predecessor of scheme nodes will be the premise and successor of the RA-node and will be claim of the argument. Similarly, CA-nodes and PA-nodes are translated into counter-argument and defeat the relationship between arguments respectively. The blank-nodes are translated as sub-argument

[6] For simplicity we have not mentioned the identifers of the arguments.

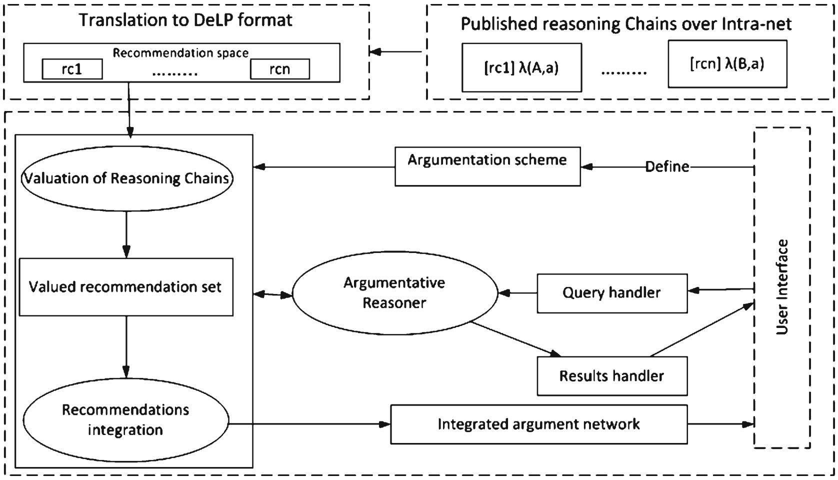

Translation to DeLP format
Recommendation space
rc1
....
rcn
Published reasoning Chains over Intra-net
[rc1] λ(A,a)
....
[rcn] λ(B,a)
Valuation of Reasoning Chains
Argumentation scheme
Define
Valued recommendation set
Argumentative Reasoner
Query handler
User Interface
Results handler
Recommendations integration
Integrated argument network

Fig. 9 Conceptual framework of Web@KIDSS

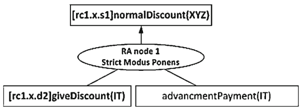
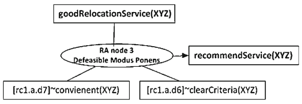
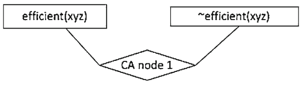
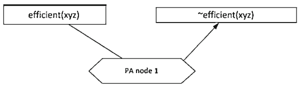
relationship between arguments. Some examples of the translation are depicted in Table 2 below.

7.2 Definition of integration scheme

Once the task of translating the reasoning chains has been accomplished, the next step is to define the integration scheme. The integration scheme, derived from the concept of argumentation scheme, corresponds to our daily life pattern of reasoning. To further explain it, let us consider the case study discussed in Section 3 where higher authorities has a set of recommendations and wants to select only those recommendations that satisfy certain specific criteria. For example, higher authorities specify criteria that the recommendation must be provided for relocation service provider XYZ. Therefore, only recommendations for XYZ are considered for the final decision-making process. The scope of application of an integration scheme ranges from valuation of reasoning chains and their integration during the decision-making process. In the proposed framework, DeLP language is used to create an integration scheme using the following steps:

1. Enter name of integration scheme.
2. Define set of premises.
3. Define set of critical questions. The critical questions are queries to be executed on a reasoning chain. The critical questions are further categorised as follows:
   (a) Set of assumptions
   (b) Set of exceptions

Table 2 Translation of AIF compliant reasoning chain in Web@KIDSS

Scheme Node	AIF representation	Translation
Strict inference	 [rc1.x.s1]normalDiscount(XYZ) RA node 1 Strict Modus Ponens [rc1.x.d2]giveDiscount(IT) advancmentPayment(IT)	If the RA-Nodes use strict modus ponens, then all the incoming edges to RA-Node are considered as premises and the successor node is considered as claim of strict argument.
Defeasible inference	 goodRelocationService(XYZ) RA node 3 Defeasible Modus Ponens recommendService(XYZ) [rc1.a.d7]~convienent(XYZ) [rc1.a.d6]~clearCriteria(XYZ)	If the RA-Nodes use defeasible modus ponens, then all the incoming edges to RA-Node are considered as premises and the successor node is considered as claim of defeasible argument
CA-node		No translation, as proposed system has a built-in mechanism to identify conflicting arguments
PA-node		No translation, as proposed system has a built-in mechanism to identify priority among conflicting arguments.

4. Set conflict handling variant i.e. conflict blocking either true or false. The scope of conflict handling can be defined at valuation of reasoning chains or their integration or at both levels.
   (a) During valuation of a reasoning chain, if there exists any conflict between the a critical question and the premise, then in the case of conflict blocking variant true, the reasoning chain is not considered suitable for knowledge integration and vice versa.
   (b) During knowledge integration, if there exists a conflict between two arguments coming from different reasoning chains, then in the case of conflict blocking variant true, those arguments are not considered in the final decision-making process and vice versa.

Table 3 below depicts the definition of an integration scheme for knowledge integration.

7.3 Valued recommendation set

After defining the integration scheme, now the system applies the user-defined integration scheme to each reasoning chain. This process requires the following two steps:

1. Modelling of reasoning chains
2. Application of integration scheme to reasoning chains

7.3.1 Modelling of reasoning chains

The system first models the reasoning chain by identifying its basic elements as determined by Toulmin (2003). A reasoning chain is modelled as follows:

1. **Back-up evidence**: The initial working memory describing the current situation, from which the argumentative reasoner starts its derivation activity. In a reasoning chain, these nodes have no incoming edge (no predecessor nodes) and only an outer edge, or successor nodes, are considered as back-up evidence.
2. **Claim**: The result of the reasoning chain corresponds to claim.
3. **Warrant**: The support for the result of a reasoning chain is called a warrant. It is a set of arguments linked up to form a reasoning chain link as back-up evidence for a claim.

Such modelling of a reasoning chain has a significant relevance for correctly modelling a practical argumentation activity and helps to categorize the various ways by which arguments can be analysed and defeated and therefore the following strategies could have significant value as identified by Baroni et al. (1998). If conflict exists between a critical question and data, then the entire conclusion drawn from them is undermined. Similarly, it could help to point out flaws in the reasoning chain that relate data to the conclusion. Additionally, if conflict exists between claim and critical question, then the decision maker has to see the warrant and data in order to defeat the claim.

7.3.2 Application of integration scheme on reasoning chains

After the modelling of reasoning chains, the system applies the integration scheme defined by the system user to each and every reasoning chain. This involves executing all commands against the selected reasoning chain as depicted in Table 3. During this process, if any conflict exists either between data and premise, or conflict between critical question and warrant, then the system stores those results and depending upon conflict blocking variable value, the reasoning chain will be considered for the knowledge integration phase. The system also displays the results to the system user so that conflicts can be resolved if possible.

7.4 Knowledge integration

Once the valuation of reasoning chains has been completed, the next step is knowledge integration. This step involves integration of the diverse valued reasoning chains into a single consolidated reasoning chain to provide a complete picture to the decision maker to

Table 3 Integration scheme mapping to DeLP

	Integration scheme	DeLP construct
Scheme name	SupplierIntegrationScheme	
Premise	– The recommendations are against Relocation Service provider XYZ	– Execute(relocationService(xyz))
Critical questions	– The XYZ is good at formalising the clients' criteria	– Execute(clearCriteria(xyz))
Variant	– Conflict blocking is true for valuation of reasoning chains	– Conflict-blocking=true

support the decision-making process. This step comprises the following tasks:

1. Identification of conflicts among arguments belonging to different valued reasoning chains in a valued recommendation set.
2. Automated resolution of conflicts between arguments with the help of static and dynamic defeat. In case of blocking arguments, the system needs human intervention to resolve the conflict between them.
3. Construction of new arguments. If two arguments from a valued recommendation set have the same claim, then combine the premises of those arguments to produce a new argument.
4. Building up of reasoning chains and providing an interface to system user to making a final judgement as depicted in Fig. 10

7.5 Query the valuated set

The system also provides an interface to query the valuated reasoning chains.

Definition 13 (Query) A query 'q' , consists of a predicate, and can be executed on the argument set Args with the help of function executeQuery(q) $\in \mathcal{F}$ to check the support for the predicate in the recommendation space.

There are four possible answers to a query, as follows:

- If the answer is 'yes', then the result will be an undefeated dialectical tree. Mathematically, it is presented as follows:

$\Sigma_U(\mathcal{A}, h) = \mathit{executeQuery}(q)$.......... Equation (24)

- If the answer is 'no', then the result will be a defeated dialectical tree. Mathematically, it is presented as follows:

$\Sigma_D(\mathcal{A}, h) = \mathit{executeQuery}(q)$.......... Equation (25)

- If the answer is 'undecided', then the result will be a blocked dialectical tree. Mathematically, it is presented as follows:

$\Sigma_B(\mathcal{A}, h) = \mathit{executeQuery}(q)$.......... Equation (26)

- Unknown, if the predicate in the query is not in the language of the program. Mathematically, it is presented as follows:

$\mathit{unknown} = \mathit{executeQuery}(q)$......... Equation (27)

7.6 User interface

The user interface is the graphical representation of a reasoning engine output for the end user. The user interface component will be responsible for representing the argumentation process and justifies conclusions to the user in the form of an inverted tree-like structure, and the user will be able to interact with and query the results.

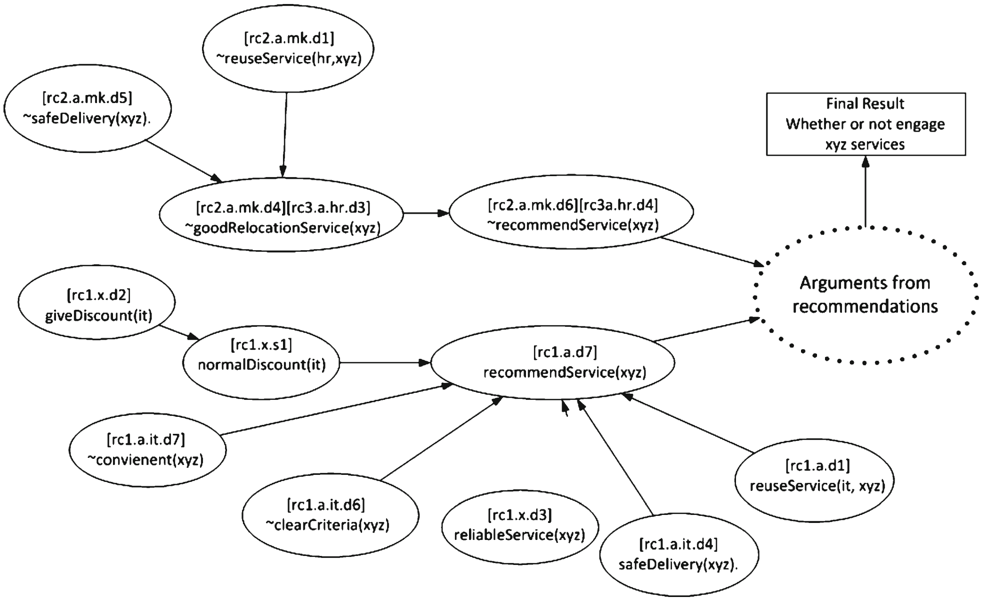


Fig. 10 Graphical representation of integrated knowledge for decision support

7.7 Algorithms for knowledge integration

In this section, we describe the working of our knowledge integration algorithms. Algorithm 1 (knowledge integration) invokes Algorithm 2 (valuation of a reasoning chain) and Algorithm 3 (combine reasoning chains). Algorithm 1 takes into account a set of AIF-compliant reasoning chains and sets their valuation flag to false. Then, the system applies the user-defined integration scheme to each of the reasoning chains one by one; this is known as the valuation of a reasoning chain. This step is carried out by invoking Algorithm 2. Algorithm 2 takes into account a single reasoning chain and a user-defined integration scheme such as *supplierIntegrationScheme* shown in Table 3. During valuation, all the queries generated through the integration scheme are executed on a reasoning chain. If the result of a query execution is false, this will establish a conflict between the reasoning chain content and the integration scheme content. The algorithm returns true if there is no conflict between the integration scheme and reasoning chain or the conflict blocking flag has a value of false. Otherwise, this algorithm will return false. The process of valuation applies to all the reasoning chains. After valuation, reasoning chains are ready for the next step: knowledge integration. All those reasoning chains with valuation flags true are considered in the knowledge integration phase. For knowledge integration, the system first integrates all those reasoning chains whose

Algorithm 1: Knowledge integration

```
Data: Recommendation space {rc1,rc2,rc3}
Result: Integrated arguments network

Array rc []= {rc1, rc2, rc3};
int i=0;
supplierIngtegrationScheme si;
System user initialize si;
foreach rc.length do
    rc[i].valued=false;
    i++;
end
foreach rchain in rc do
    boolean a = Valuation_of_reasoning_chain(rchain,si);
    if a is true then
        rc.valued = true;
    else
        rc.remove(rchain);
    end
end
i =0;
rc = combineReasoningChain(rc, sameResults);
displayResult(rc);
```

Algorithm 2: Valuation of a reasoning chain

```
Data: a reasoning chains(rc) and integration scheme(is)
Result: boolean
foreach premise in is do
    if result = execute(premise) on rc then
        if result= false then
            addconflict(premise, rc);
        end
    end
end
foreach cq in is do
    if result = execute(cq) on is then
        if result= false then
            addconflict(cq, rc);
        end
    end
end
if addconflict[] !=null or is.conflictProgation=true then
    return true;
else
    return false;
end
```

Algorithm 3: Combine reasoning chains

```
Data: Valued recommendation set, claimflag
Result: Array of combined reasoning chain

Array commonClaims [ ];
Array conflictingArguments [ ];
for i=0;i <rc.length(); i$^{++}$ do
    for j=i;j <rc.length(); j$^{++}$ do
        if flag==sameClaims then
            condition=rc[i].result ==rc[j].result
        else
            condition=rc[i].result !=rc[j].result
        end
        if condition then
            commonArgument = rc[i] $\otimes$ rc[j];
            activeArgumentSet= rc[i] $\odot$ rc[j];
            conflictingArguments = rc[i] $\oslash$ rc[j];
            foreach c in commonClaims do
                argRc1 = getArgument(c, rc1);
                argRc2 = getArguments(c, rc2);
                r = argRc1 $\boxplus$ argRc2;
                activeArgumentSet=r;
                i$^{++}$;
            end
            foreach arg in conflictingArguments do
                argRc1 = getArgument(arg, rc1);
                argRc2 = getArguments(arg, rc2);
                if conflictblocking==true then

                    activeArgumentSet = argRc1;
                    activeArgumentSet = argRc2;
                    static or dynamic defeat(argRc1, argRc2);
                    userPreference=getPreferences(argRc1,argRc2);
                    if userPreference(argRc1,argRc2) then
                    preferenceSet = argRc1 > argRc2;
                end
            end
        end
        rc [i]=BuildupReasoningChain(activeArgumentSet);
    end
end
return rc;
```

Algorithm 4: Buildup a reasoning chain

```
Data: activeArgumentSet
Result: $\lambda_{(\mathcal{A},h)}$
Let S=subArgumentsOf(($\mathcal{A}$, $h$));
if $S \neq \emptyset$ then
    foreach argument ($\mathcal{A}_i$, $h_i$) $\in S$ do
        if noCounterArgument($\mathcal{A}_i$, $h_i$) then
            BuildReasoningChain(($\mathcal{A}_i$, $h_i$)) ;
            Put $\lambda$(($\mathcal{A}_i$, $h_i$))as a immediate supporting argument of ($\mathcal{A}$, $h$);
        end
    end
else
    $\lambda_{(\mathcal{A},h)}$ = ($\mathcal{A}$, $h$);
end
```

results are the same or support the same point of view. This task is performed by invoking Algorithm 3 with a set of valuated reasoning chains. Algorithm 3 first loops through a set of reasoning chains and compares the result of a reasoning chain with the result of the remaining reasoning chains; if the results match, then those reasoning chains are integrated. Three kinds of operators are used during this integration process. With the help of a focus operator ($\otimes$), the new arguments are constructed and then loaded into a valued recommendation set. With the help of a unique operator ($\odot$), unique arguments from both reasoning chains are loaded into an valued recommendation set. With the help of the conflict operator ($\oslash$) the conflicting arguments are taken into account for conflict resolution. If the conflict blocking flag for knowledge integration is false, then the system tries to resolve conflicts with the help of static or dynamic defeat. Otherwise, the system asks the end user to choose between the conflicting arguments. Finally, Algorithm 3 invokes Algorithm 4 (Buildup a reasoning chain) in order to establish a reasoning chain from the argument loaded in the valued recommendation set. The important thing to note here is that conflicts may exist in a valued recommendation set if the conflict blocking flag is true. The display function in Algorithm 1 displays the integrated reasoning chains to the user as depicted in Fig. 10. The integrated reasoning chains depict the different points of view supported by the set of arguments. The end user can make a decision based on the integrated information. The system saves the decision and makes it available for future reference.

8 Implementation and prototype development

This section provides the implementation details and working of Web@KIDSS to represent and reason over incomplete and conflicting recommendations coming from different departments. The development of the prototype system is carried out on a machine having an Apache Web server version 2.2.11, PHP version 5.3.0, PHP Tree Graph Ext library[7] with certain extensions to differentiate between fact and claim of a rule, strict and defeasible inference etc, MySql database version 5.1.36 and SWI-Prolog installed on it. After prototype development, the Web application is deployed on the DEBII server.

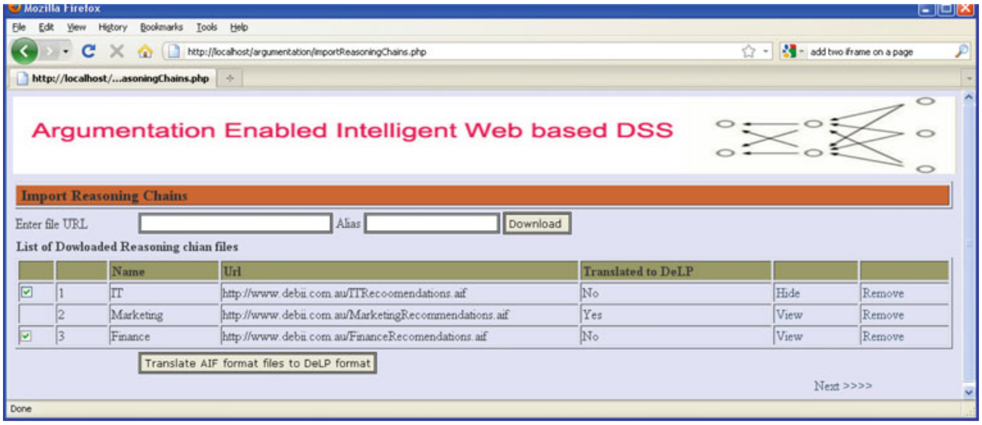
Figure 11 shows the interface provided to the system user to import the recommendations published in the form of reasoning chains in AIF format over the enterprise intranet. The user can download the file by entering the URL and name in the provided text fields and click the ‘Download button’. The interface also shows the list of downloaded AIF compliant reasoning chain files and the user is able to either view or remove them from the Web@KIDSS. Once the user has finished downloading the recommendation files, s/he can then translate the downloaded files into DeLP format. Figure 11 depicts the interface where the user can select the files by clicking the check-boxes and submitting the selected files for translation by clicking the ‘Translate AIF format files to DeLP’ format button.

Once the user has finished importing the AIF files, s/he can then define an integration scheme for the valuation of reasoning chains. Figure 12 depicts an interface where a user can define premises that need to be matched, queries to be executed, and conflict blocking variant at valuation of a reasoning chain and knowledge integration levels. The end user also gives the integration scheme a name. Once the user has finished the integration scheme, s/he proceeds to the next step which is the valuation of reasoning chains by clicking the ‘Next’ link (shown in Fig. 12 above). Figure 13 depicts the interface where the user can select reasoning chains and click the ‘Apply Integration Scheme’ button to trigger the valuation process.

Once the user clicks the Apply Integration Scheme button, the Web@KIDSS applies all the premises that need to be matched and queries to be executed on reasoning chains. The user can view the outcome of the valuation process by clicking on the ‘View’ link against the valuated reasoning chain as depicted in Fig. 14. The text in red shows the conflict between the integration scheme and the contents of a reasoning chain.

After the valuation of reasoning chains, the next step is knowledge integration whereby all the reasoning chains are integrated to depict the overall problem. Figure 15 depicts the knowledge integration under two arguments: recommend the supplier XYZ as preferred

[7] http://download.getabest.com/new/php-tree-graph-ext-222943.html


Mozilla Firefox
File Edit View History Bookmarks Tools Help
http://localhost/argumentation/importReasoningChains.php
add two iframe on a page
http://localhost/...asoningChains.php
Argumentation Enabled Intelligent Web based DSS

Import Reasoning Chains

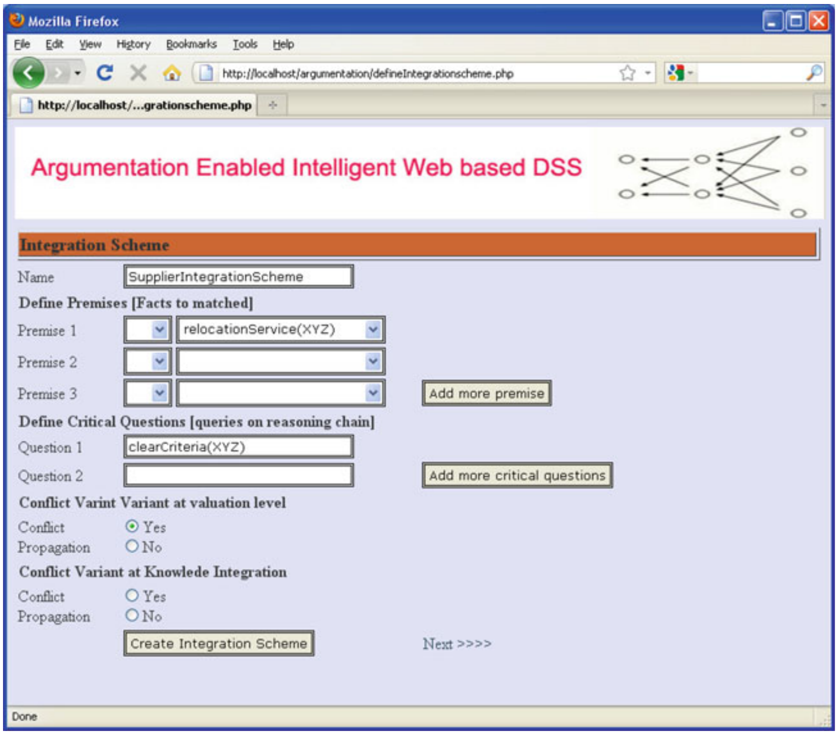
Enter file URL ____ Alias ____ Download

List of Dowloaded Reasoning chian files

		Name	Url	Translated to DeLP		
☑	1	IT	http://www.debii.com.au/ITRecoomendations.aif	No	Hide	Remove
	2	Marketing	http://www.debii.com.au/MarketingRecommendations.aif	Yes	View	Remove
☑	3	Finance	http://www.debii.com.au/FinanceRecomendations.aif	No	View	Remove

Translate AIF format files to DeLP format
Next >>>>
Done

Fig. 11 Interface to import reasoning chains


Mozilla Firefox
File Edit View History Bookmarks Tools Help
http://localhost/argumentation/defineIntegrationscheme.php
http://localhost/...grationscheme.php
Argumentation Enabled Intelligent Web based DSS

Integration Scheme

Name: SupplierIntegrationScheme

Define Premises [Facts to matched]

Premise 1: relocationService(XYZ)

Premise 2:

Premise 3:

Define Critical Questions [queries on reasoning chain]

Question 1: clearCriteria(XYZ)

Question 2:

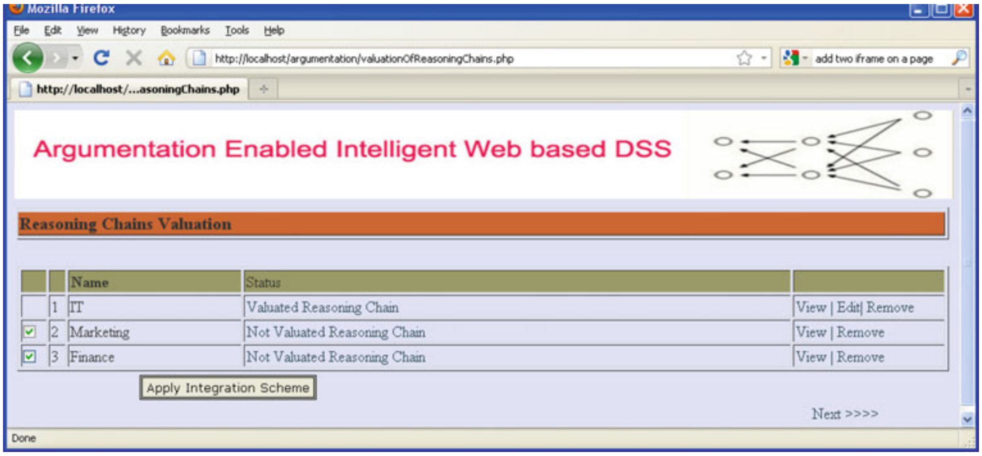
Conflict Varint Variant at valuation level

Conflict Propagation: ● Yes ○ No

Conflict Variant at Knowlede Integration

Conflict Propagation: ○ Yes ○ No

Fig. 12 Interface to define integration scheme

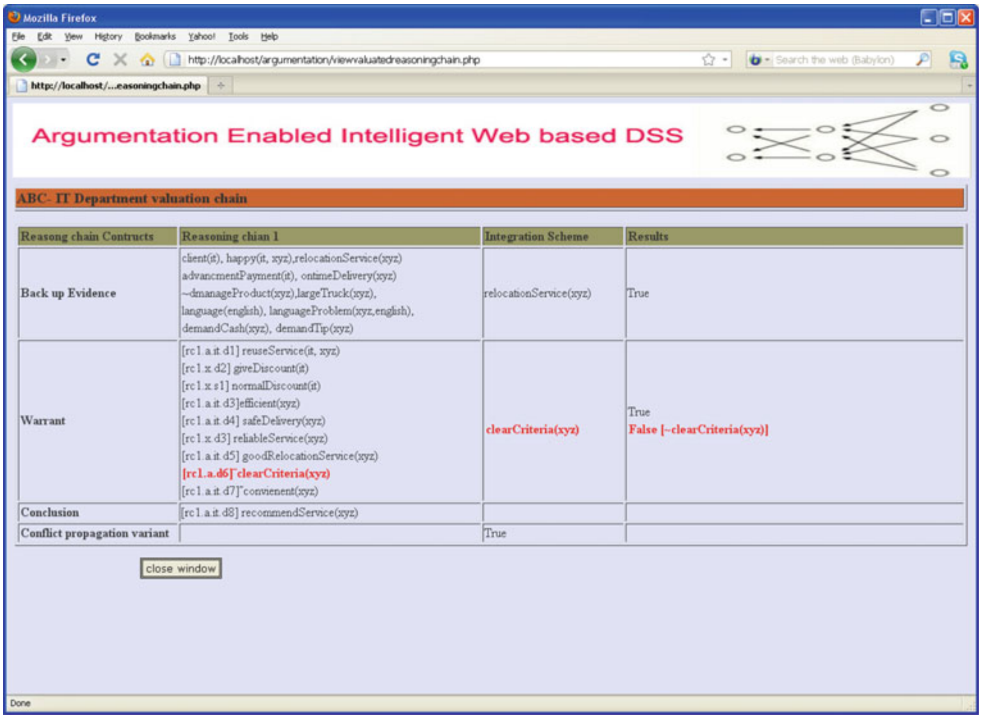

Mozilla Firefox
File Edit View History Bookmarks Tools Help
http://localhost/argumentation/valuationOfReasoningChains.php
add two iframe on a page
http://localhost/...asoningChains.php
Argumentation Enabled Intelligent Web based DSS

Reasoning Chains Valuation

		Name	Status	
	1	IT	Valuated Reasoning Chain	View | Edit| Remove
☑	2	Marketing	Not Valuated Reasoning Chain	View | Remove
☑	3	Finance	Not Valuated Reasoning Chain	View | Remove

Apply Integration Scheme
Next >>>>
Done

Fig. 13 Interface to select reasoning chains and apply integration scheme


Mozilla Firefox
File Edit View History Bookmarks Yahoo! Tools Help
http://localhost/argumentation/viewvaluatedreasoningchain.php
Search the web (Babylon)
http://localhost/...easoningchain.php
Done

Argumentation Enabled Intelligent Web based DSS

ABC- IT Department valuation chain

Reasong chain Contructs	Reasoning chian 1	Integration Scheme	Results

Back up Evidence	client(it), happy(it, xyz),relocationService(xyz)
	advancmentPayment(it), ontimeDelivery(xyz)
	~dmanageProduct(xyz),largeTruck(xyz),
	language(english), languageProblem(xyz,english),
	demandCash(xyz), demandTip(xyz)
		relocationService(xyz)	True

Warrant	[rc1.a.it.d1] reuseService(it, xyz) [rc1.x.d2] giveDiscount(it) [rc1.x.s1] normalDiscount(it) [rc1.a.it.d3]efficient(xyz) [rc1.a.it.d4] safeDelivery(xyz) [rc1.x.d3] reliableService(xyz) [rc1.a.it.d5] goodRelocationService(xyz) **[rc1.a.d6]˜clearCriteria(xyz)** [rc1.a.it.d7]˜convienent(xyz)	**clearCriteria(xyz)**	True **False [~clearCriteria(xyz)]**
Conclusion	[rc1.a.it.d8] recommendService(xyz)		
Conflict propagation variant		True	

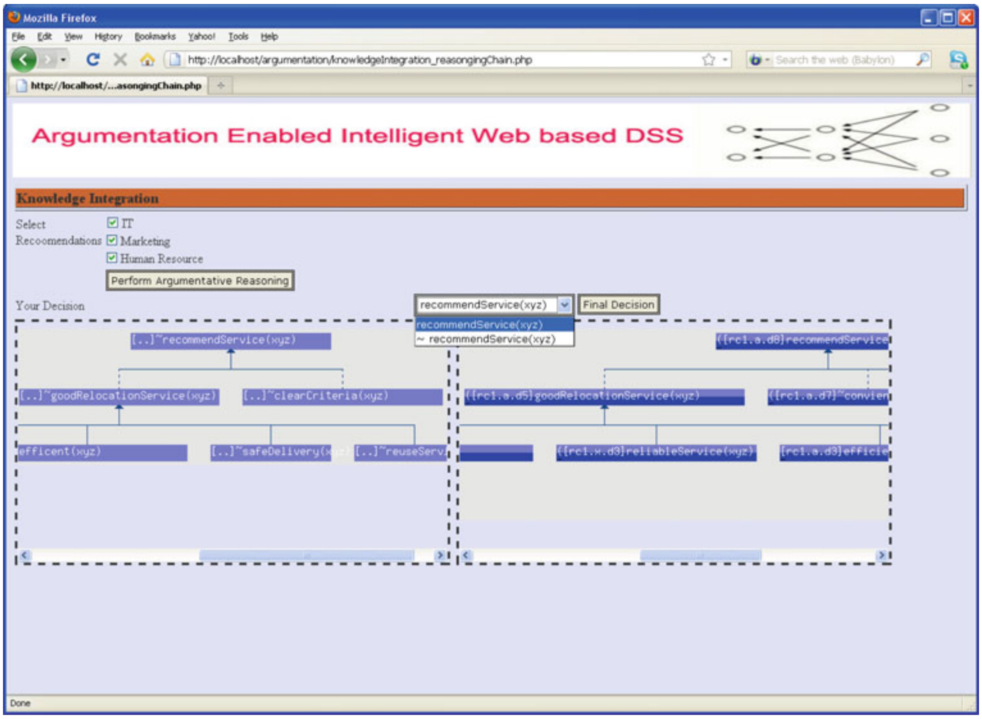
close window

Fig. 14 Interface depicting the result of a reasoning chain valuation

client or not recommend it as preferred client. The final decision needs to be made by the end user who selects the result from the drop-down menu and clicks the 'Final Decision' button. This will save the system user's preference in the knowledge base.

9 Conclusion and future directions

In this article, we have presented a solution for enterprise-wide information and knowledge integration for intelligent decision making. We pointed out


Mozilla Firefox
File Edit View History Bookmarks Yahoo! Tools Help
http://localhost/argumentation/knowledgeIntegration_reasongingChain.php
Search the web (Babylon)
http://localhost/...asongingChain.php
[..]˜recommendService(xyz)
[..]˜goodRelocationService(xyz)
[..]˜clearCriteria(xyz)
efficent(xyz)
[..]˜safeDelivery(xyz)
[..]˜reuseServ
([rc1.a.d8]recommendService
([rc1.a.d5]goodRelocationService(xyz)
([rc1.a.d7]˜convien
([rc1.x.d3]reliableService(xyz)
[rc1.a.d3]efficie
Done

Argumentation Enabled Intelligent Web based DSS

Knowledge Integration

Select Recoomendations ☑ IT
☑ Marketing
☑ Human Resource

Perform Argumentative Reasoning

Your Decision recommendService(xyz) Final Decision
recommendService(xyz)
~ recommendService(xyz)

Fig. 15 Interface presenting integrated knowledge to facilitate final decision

that the semantic web, while it addresses the issues of syntactical and semantical heterogeneity of information to integrate and benefit the decision support system, nevertheless does not address the issue of incomplete and conflicting information integration. Several researchers have attempted to address this issue, but their efforts have provided a formalism to represent and handle only individual preferences in the form of priorities among the conflicting rules. However, DSS systems are subject to inconsistencies deriving from multiple sources and multiple users; therefore, it is not possible to define priorities in advance in order to resolve conflict among rules derived from multiple sources/users. This limitation of the current Web-DSS also prevents enterprises from being able to integrate knowledge.

In this article, we extend our previous work on Web@IDSS to make its results shareable in AIF format. We also provide formal syntax and semantics for Web@KIDSS. The Web@KIDSS is equipped with argumentative reasoning and an argumentation scheme for knowledge integration. Therefore, Web@KIDSS is capable of handling incomplete and contradictory knowledge in the form of reasoning chains published over the web or enterprise intranet by diverse Web-DSS or Web@IDSS. The major contributions of this article are as follows:

1. Extension to Web@IDSS with AIF reification resulting in sharing of results in the form of AIF-compliant reasoning chains.
2. Formalization of syntax and semantics for knowledge integration in Web@KIDSS.
3. A proposed conceptual framework for representing, reasoning and integrating incomplete and conflicting reasoning chains for knowledge integration in Web@KIDSS.
4. Design and development of algorithms for knowledge integration and their validation through prototype development.

Our future work will be primarily along the following lines:

1. Enterprise environments are becoming increasingly complex, competitive and dynamic. The business policies change dynamically and frequently to keep pace with the competitive nature of business environments. However the actual processes carried out in day-to-day business environments are not always in consonance with the new business policies (Wang et al. 2009). This situation is more profound in the case of managing dynamic processes where environment changes rapidly (Pesic and van der Aalst 2006). This demands for an enterprise business process modeling methodology that automatically builds models and executes task specific models in response to user queries (Ba et al. 1997). Such an approach should be flexible enough for e-Collaboration for business process modeling amongst different participants to address new challenges such as business process mergers. To address above mentioned challenge, we aim to design and develop policy-centric information system by extending the argumentation based intelligent decision making techniques proposed in this paper. We also aim at to introduce a graphical language to represent different process constructs and their linkages in a process model.
2. In the past decade or so, numerous machine learning methods have been used to automatically learn and recognize complex patterns and make intelligent decisions based on enterprise data. One of the common attributes of these machine leaning methods is that their working and functionality is constrained by the amount of input data. However, the scale of the enterprise data has increased mani-fold (leading to the concept of Big Data), thereby in many cases rendering the underlying machine learning algorithms either incapable of managing such large and ever increasing data, or too slow for decision making. In our further work we intend to enhance the current generation of machine learning techniques with argumentation formalisms described in this paper. In such cases, the arguments from experts are considered during mining of enterprise data.[8] Such work will lay down foundations for performing large-scale analytics on big data in an enterprise. Such an approach would make use of cloud platforms.

References

Alaranta, M., & Henningsson S. (2008). An approach to analyzing and planning post-merger is integration: insights from two field studies. *Information Systems Frontiers, 10*, 307–319.

Antoniou, G., & Bikakis, A. (2007). Dr-prolog: a system for defeasible reasoning with rules and ontologies on the semantic web. *IEEE Transactions on Knowledge and Data Engineering, 19*(2), 233.

Antoniou, G., Baldoni, M., Bonatti, P.A., Nejdl, W., Olmedilla, D. (2004). Rule-based policy specification. In Yu, T., Jajodia, S. (Eds.), *Secure data management in decentralized systems* (pp. 169–216, Vol. 33). US: Springer

Antoniou, G., Damasio, C.V., Grosof, B., Horrocks, I., Kifer, M., Maluszynski, J., Patel-Schneider, P.F. (2005).

[8] http://www.ailab.si/martin/abml/

Combining rules and ontologies: A survey. Tech. Rep. IST-2004-506779 REWERSE Deliverable I3-D3, Technical Report IST506779/Linköping/I3-D3/D/PU/a1. Linköping University

Ba, S., Lang, K.R., Whinston, A.B. (1997). Enterprise decision support using intranet technology. *Decision Support Systems, 20*(2), 99–134. doi:10.1016/S0167-9236(96)00068-1, http://www.sciencedirect.com/science/article/pii/S0167923696000681.

Baroni, P., Fogli, D., Guida, G. (1998). Modeling argumentation in practical reasoning: a conceptual analysis of argument life cycle. In *7th international conference on information processing and management of uncertainty in knowledge-based systems* (pp. 1790–1797). Paris.

Bassiliades, N., Antoniou, G., Vlahavas, I. (2004). Dr-device: A defeasible logic system for the semantic web. *Principles and practice of semantic web reasoning* (pp 134–148).

Benkö, T., Lukácsy, G., Fokt, A., Szeredi, P. (2003). Information integration through reasoning on meta-data. In *AI moves to IA: Workshop on artificial intelligence, information access, and mobile computing*.

Berners-Lee, T. (2000). Semantic web - xml2000, W3C. http://www.w3.org/2000/Talks/1206-xml2k-tbl/. Accessed 1 March 2012

Bex, F., Prakken, H., Reed, C. (2010). A formal analysis of the aif in terms of the aspic framework. In *3rd international conference on computational models of argument*.

Bhatt, M., Flahive, A., Wouters, C., Rahayu, W., Taniar, D. (2006). Move: a distributed framework for materialized ontology view extraction. *Algorithmica, 45*, 457–481. doi:10.1007/s00453-006-1221-2, http://portal.acm.org/citation.cfm?id=1165166.1165175.

Buccella, A., Cechich, A., Fillottrani, P. (2009). Ontology-driven geographic information integration: a survey of current approaches. *Computers & Geosciences 35*(4), 710–723. Geoscience Knowledge Representation in Cyberinfrastructure.

Carlsson, C, & Turban, E. (2002). Dss: directions for the next decade. *Decision Support Systems, 33*(2), 105–110.

Ceccaroni, L., Cortés, U., Sànchez-Marrè, M. (2004). Ontowedss: augmenting environmental decision-support systems with ontologies. *Environmental Modelling & Software, 19*(9), 785–797. Environmental Sciences and Artificial Intelligence.

Chesnevar, C., McGinnis, J., Modgil, S., Rahwan, I., Reed, C., Simari, G., South, M., Vreeswijk, G., Willmott, S. (2006a). Towards an argument interchange format. *The Knowledge Engineering Review, 21*(4), 293.

Chesnevar, C.I., Maguitman, A.G., Simari, G.R. (2006b). Argument-based critics and recommenders: a qualitative perspective on user support systems. *Data & Knowledge Engineering, 59*(2), 293.

Cheung, K., & Cheong, M.P. (2007). Intelligent on-line decision support tools for market operators. In *International conference on intelligent systems applications to power systems, 2007* (pp. 1–6).

Chua, W.W.K., & Goh, A.E.S. (2010). Techniques for discovering correspondences between ontologies. *International Journal of Web and Grid Services Archive, 6*(3), 213–243. doi:10.1504/IJWGS.2010.035090.

Fan, X., Toni, F., Hussain, A. (2010). Two-agent conflict resolution with assumption-based argumentation. In *Proceeding of the 2010 conference on computational models of argument: Proceedings of COMMA 2010* (pp. 231–242). Amsterdam, The Netherlands: IOS Press.

Flahive, A., Rahayu, W., Taniar, D., Apduhan, B. (2005). A distributed ontology framework in the semantic grid environment. In *19th international conference on Advanced Information Networking and Applications, 2005. AINA 2005* (pp. 193–196, Vol. 2). doi:10.1109/AINA.2005.19.

Flahive, A., Taniar, D., Rahayu, W., Apduhan, B.O. (2009). Ontology tailoring in the semantic grid. *Computer Standards & Interfaces, 31*(5), 870–885. doi:10.1016/j.csi.2008.03.016, http://www.sciencedirect.com/science/article/pii/S0920548908000330. Specification, Standards and Information Management for Distributed Systems.

Garcia, A.J., & Simari, G.R. (2004). Defeasible logic programming: an argumentative approach. *Theory and Practice of Logic Programming, 4*(1+2), 95–138.

Garcia-Crespo, A., Ruiz-Mezcua, B., Lopez-Cuadrado, J.L., Gonzalez-Carrasco, I. (2011). Semantic model for knowledge representation in e-business. *Knowledge-Based Systems, 24*(2), 282–296. doi:10.1016/j.knosys.2010.09.006.

Godden, D.J., & Walton, D. (2007) Advances in the theory of argumentation schemes and critical questions. *Informal Logic, 27*, 267–292.

Grosof BN, Gandhe MD, Finin TW (2002) Sweetjess: Translating damlruleml to jess. In *Proceedings of the international workshop on rule markup languages for business rules on the semantic web*.

Hurt, C.D. (1998). Nonmonotonic logic for use in information retrieval: an exploratory paper. *Information processing & management, 34*(1), 35.

Iyad Rahwan, C.R. (2009). *The argument interchange format, argumentation in artifical intelligence*. Springer.

Janjua, N.K., & Hussain, F.K. (2011). Web@idss : argumentation-enabled web-based idss for reasoning over incomplete and conflicting information. *Knowledge-Based Systems, 32*, 9–27. doi:10.1016/j.knosys.2011.09.009, http://www.sciencedirect.com/science/article/pii/S0950705111002103.

Kartha, N., & Novstrup, A. (2009). Ontology and rule based knowledge representation for situation management and decision support. In Mott, S., Buford, J.F., Jakobson, G., Mendenhall, J.M. (Eds.), *Intelligent sensing, situation management, impact assessment, and cyber-sensing*. SPIE.

Katie Atkinson, T.B.C. (2008). *Abstract argumentation scheme frameworks, artificial intelligence: Methodology, systems, and applications* (Vol. 5253/2008). Berlin/Heidelberg: Springer.

Kim, J., Kim, P., Chung, H. (2011). Ontology construction using online ontologies based on selection, mapping and merging. *International Journal of Web and Grid Services, 7*, 170–189.

Kontopoulos, E., Bassiliades, N., Antoniou, G. (2011). Visualizing semantic web proofs of defeasible logic in the DR-DEVICE system. *Knowledge-Based Systems, 24*(3), 406–419. doi:10.1016/j.knosys.2010.12.001.

Lee, T.B. (2003). The semantic web and challenges. W3C. http://www.w3.org/2003/Talks/01-sweb-tbl/. Accessed 1 March 2012.

Lee, T.B. (2005). Www 2005 keynote. W3C. http://www.w3.org/2005/Talks/0511-keynote-tbl/. Accessed 1 March 2012.

Lee, T.B. (2006). Artificial intelligence and the semantic web: Aaai 2006 keynote. W3C. http://www.w3.org/2006/Talks/0718-aaai-tbl/Overview.html. Accessed 1 March 2012.

Letia, I., & Groza, A. (2008). A planning-based approach for enacting world wide argument web. In Badica, C., Mangioni, G., Carchiolo, V., Burdescu, D. (Eds.), *Intelligent distributed computing, systems and applications, studies in computational intelligence* (pp. 137–146, Vol. 162). Berlin/Heidelberg: Springer.

Liu, S., Duffy, A., Whitfield, R., Boyle, I. (2010). Integration of decision support systems to improve decision support performance. *Knowledge and Information Systems, 22*, 261–286.

Loui, R.P. (1998). Process and policy: Resource-bounded nondemonstrative reasoning. *Computational intelligence, 14*(1), 1.

March, S.T., & Hevner, A.R. (2007). Integrated decision support systems: a data warehousing perspective. *Decision Support Systems, 43*(3), 1031–1043. Integrated Decision Support.

Morge, M. (2008). The hedgehog and the fox: An argumentation-based decision support system. In: *Proceedings of the 4th international conference on Argumentation in Multi-agent Systems, ArgMAS'07* (pp. 114–131). Berlin, Heidelberg: Springer-Verlag.

Muthaiyah, S., & Kerschberg, L. (2007). Virtual organization security policies: an ontology-based integration approach. *Information Systems Frontiers, 9*, 505–514. doi:10.1007/s10796-007-9050-7.

Nguyen, H.Q., Taniar, D., Rahayu, J.W., Nguyen, K. (2011). Double-layered schema integration of heterogeneous xml sources. *Journal of Systems and Software, 84*(1), 63–76. doi:10.1016/j.jss.2010.07.055. Information Networking and Software Services.

Nicolicin-Georgescu, V., Benatier, V., Lehn, R., Briand, H. (2010). Ontology-based autonomic computing for decision support systems management: Shared ressources allocation between groups of data warehouses. In *2010 3rd international conference on Communication Theory, Reliability, and Quality of Service (CTRQ)* (pp. 233–236). doi:10.1109/CTRQ.2010.46.

Norta, A., & Eshuis, R. (2010). Specification and verification of harmonized business-process collaborations. *Information Systems Frontiers, 12*, 457–479. doi:10.1007/s10796-009-9164-1.

Noy, N.F. (2004). Semantic integration: a survey of ontology-based approaches. *SIGMOD Record, 33*, 65–70.

Osei-Bryson, K.M., & Ngwenyama, O. (2008). Decision models for information systems management. *Information Systems Frontiers, 10*, 277–279.

Palau, R.M., & Moens, M.F. (2009). Argumentation mining: the detection, classification and structure of arguments in text. In *ICAIL '09: Proceedings of the 12th international conference on artificial intelligence and law* (pp. 98–107). New York, NY: ACM.

Pesic, M., & van der Aalst, W. (2006). A declarative approach for flexible business processes management. In Eder, J., & Dustdar, S. (Eds.), *Business process management workshops, lecture notes in computer science* (Vol. 4103, pp 169–180). Berlin/Heidelberg: Springer. doi:10.1007/11837862_18.

Power, D.J. (2002). *Decision support systems: Concepts and resources for managers*. Greenwood Publishing Group.

Power, D.J., & Sharda, R. (2009). Decision support systems. In Nof, S.Y. (Ed.), *Springer handbook of automation* (pp. 1539–1548). Berlin/Heidelberg: Springer.

Rahwan, I., Zablith, F., Reed, C. (2007a). Towards large scale argumentation support on the semantic web. In *AAAI'07: Proceedings of the 22nd national conference on artificial intelligence* (pp. 1446–1451). AAAI Press.

Rahwan, I., Zablitha, F., Reed, C. (2007b). Laying the foundations for a world wide argument web. *Artificial intelligence, 171*(10–15), 897.

Salam, A. (2007). Design and implementation of semantic decision support system for supplier performance contract monitoring and execution: Integrating description logics, semantic web rules and service-oriented computing in the context of the extended enterprise. In *Americas conference on information systems*.

Seng, J.L., & Kong, I.L. (2009). A schema and ontology-aided intelligent information integration. *Expert Systems with Applications, 36*, 10,538–10,550.

Shim, J.P., Warkentin, M., Courtney, J.F., Power, D.J., Sharda, R., Carlsson, C. (2002). Past, present, and future of decision support technology. *Decision Support Systems, 33*(2), 111–126.

Silverman, B.G., Bachann, M., Al-Akharas, K. (2001). Implications of buyer decision theory for design of e-commerce websites. *International Journal of Human-Computer Studies, 55*(5), 815–844.

Suguri, H., Ahmad, H.F., Pasha, M., Khalid, N. (2008). *Foundation for autonomous semantic grid*. USA: Nova Science Publications.

Toni, F. (2007). E-business in argugrid. In Veit, D., & Altmann, J. (Eds.), Grid economics and business models, lecture notes in computer science (Vol. 4685, pp. 164–169). Berlin/Heidelberg: Springer.

Torroni, P., Gavanelli, M., Chesani, F. (2009). *Arguing on the semantic grid*. USA: Springer.

Toulmin, S.E. (2003). *The uses of argument*. Cambridge University Press.

Vahidov, R., Kersten, G.E. (2004). Decision station: situating decision support systems. *Decision Support Systems, 38*(2), 283–303.

Walton, D. (2009). *Argumentation in artificial intelligence, chap argumentation theory: A very short introduction* (pp. 1–24). Springer.

Wang, H.J., Zhao, J.L., Zhang, L.J. (2009). Policy-driven process mapping (pdpm): discovering process models from business policies. *Decision Support Systems, 48*(1), 267–281. doi:10.1016/j.dss.2009.08.006, http://www.sciencedirect.com/science/article/pii/S0167923609002012, Information Product Markets.

Xue, Y., Ghenniwa, H., Shen, W. (2009). Ontological view-driven semantic integration in collaborative networks. In Camarinha-Matos, L., Paraskakis, I., Afsarmanesh, H. (Eds.), *Leveraging knowledge for innovation in collaborative networks*, *IFIP advances in information and communication technology* (Vol. 307, pp. 311–318). Boston: Springer.

Yang, X., Bo, Z., Bei, Z. (2009). Research on semantic decision support system. In *WRI World congress on computer science and information engineering, 2009* (Vol. 5, pp. 687–691). doi:10.1109/CSIE.2009.364.

Zarefsky, D. (2009). *Argumentation: The study of effective reasoning* (2nd Edn., Vol. 2009). Northwestern University. URL: http://www.teach12.com/ttcx/CourseDescLong2.aspx?cid=4294.

Zhou, J., Yang, H., Wang, M., Zhang, R., Yue, T., Zhang, S., Mo, R. (2010). A survey of semantic enterprise information integration. In: *2010 3rd International Conference on Information Sciences and Interaction Sciences (ICIS)* (pp. 234–239).

Naeem Khalid Janjua is a PhD student at School of Information Systems, Curtin University, Perth, WA. He received his Master Degree in Information Technology from NUST School of Electrical Engineering and Computer Science, Islamabad, Pakistan. His areas of interests are Web-based decision support systems, Argumentation based applications, Semantic Web and Artificial Intelligence technologies.

Farookh Khadeer Hussain received the Bachelor of Technology degree in computer science and computer engineering; the M.S. degree in Information Yechnology from the La Trobe University, Melbourne, Australia; and the Ph.D. degree in Information Systems from Curtin University of Technology, Perth, Australia, in 2006. He is currently a Faculty member at School of Software, Faculty of Engineering and Information Technology, University of Technology, Sydney, Ultimo, NSW. His areas of active research are trust, reputation, trust ontologies, data modeling of public and private trust data, semantic web technologies and industrial informatics. He works actively in the domain of making informed business decisions (business intelligence) through the use of trust and reputation technology. He is interested in the application of trust and reputation as a technology, as a business analysis and intelligence tool, and the applications of trust and reputation to various domains.

Omar Khadeer Hussain received his PhD degree in Computer Science in 2008 from Curtin University. Since 2008, he is currently a Research Fellow with the School of Information Systems at Curtin University. His research interests are in the area of Risk Assessment and Management, Trusted Computing, Informed Decision Support Systems. He is a member of the IEEE and the IEEE Computer Society.