Formal Specification of Batch Scheduling Problems: A Step toward Integration and Benchmarking

Gabriela Patricia Henning

HAL Id: hal-01470607

https://inria.hal.science/hal-01470607

Submitted on 17 Feb 2017

Formal specification of batch scheduling problems: A step toward integration and benchmarking

Gabriela P. Henning

INTEC (Universidad Nacional del Litoral, CONICET), Güemes 3450, Santa Fe 3000, Argentina

`ghenning@intec.unl.edu.ar`

Abstract. This contribution presents a scheduling domain ontology, named SchedOnto, devised to tackle the formal specification of batch scheduling problems, as well as integration issues associated with the scheduling function. More specifically, this paper describes the ontological engineering approach that led to SchedOnto. The ontology characteristics along with its development process are presented, starting from the challenges that motivated the construction, the competency questions that defined the scope of the ontology, going afterwards through conceptualization and implementation stages, and finishing with some validation issues. SchedOnto relies on both, the ISA-88 and ISA-95 standards, which are well accepted in the industrial domain. After presenting SchedOnto, and its associated design process, this contribution addresses an example that shows the benefits of a formal representation of temporal aspects.

Keywords. Scheduling, Ontologies, Integration, Formal Specifications

1   Introduction

Nowadays, the importance of effective tools for supporting scheduling and planning activities in the batch process industries is undeniable. Despite the inherent difficulties of the chemical production scheduling problem [1], the academic community has recently made tremendous advances, developing efficient solution methodologies for a wide collection of problem types and plant operation scenarios. The most accepted approaches rely on a diversity of mixed-integer linear programming (MILP) models and Constraint Programming (CP) formulations. Regardless of the research done in the field, advanced scheduling support systems are not very common in the chemical industry yet [2]. In addition, most commercial systems available nowadays are not based on the many solution methodologies that academia has developed. One of the reasons why these approaches are not being used in industry is related to tool usability. Nevertheless, perhaps the most important reason for not being adopted is the fact that scheduling tools do not integrate with the other applications that are regularly employed in industrial organizations. In fact, integration of scheduling support tools with other decision support systems and transactional applications is a true challenge.

On the other hand, the ample variety of features that define a chemical production scheduling problem has led to a multiplicity of formulations that are generally oriented to better address very specific classes of problems. As a result, up to now there is no general approach that can effectively tackle the various problem classes that appear in the chemical industry [1]. Regarding evaluation, the testing of the many formulations has primarily focused on CPU requirements and is not extensive with respect to the different problem types. One key enabler for testing would be a library of case studies. However, not only such library is required, but also a way for specifying industrial-relevant case-studies in a formal, machine understandable way that can be used to automate data generation processes. Thus, there is a need for a more formal specification of scheduling problems that can help in testing and benchmarking.

This work addresses these challenges by proposing SchedOnto, a batch scheduling domain ontology. A similar approach, to tackle analogous dares that appear in the manufacturing domain, led to the PABADIS' PROMISE ontology [3]. SchedOnto has been devised to (i) tackle the formal specification of batch scheduling problems, and (ii) to serve as a foundation for the integration issues associated with the scheduling function. The contribution first describes the challenges to be faced and then presents the ontological engineering approach that led to SchedOnto. The development process is discussed in Section 3, starting from the motivation for its construction and the definition of its scope, going then through the conceptualization and implementation stages, and finishing with some future validation issues. Afterwards, a small example that shows the benefits of a scheduling domain formal specification is discussed in Section 4. It draws a distinction between the procedure function chart notation and the one proposed in SchedOnto. Finally, concluding remarks are offered.

2 Challenges to be Addressed

The scheduling activity needs to be addressed within the context of the enterprise hierarchical planning pyramid. This pyramid, which includes activities performed at different time frames, and handles information having distinct granularities, involves scheduling interplaying with the Production Planning and Control (PPC) and Plant Control (PC) functions. The difficulties associated with these interactions were pointed out almost a decade ago [4] and this topic has recently gained renewed attention. To tackle the integration of PPC and scheduling, researchers have proposed various solution strategies [5]. In addition, a few authors have pointed out which are the requirements that apply to the data exchange to support such integration [6]. Alternatively, regarding scheduling and plant control integration, researchers have started to draw the attention to data exchange issues [7], [8]. However, a more comprehensive approach is required to address these integration problems, since this matter entails much more than data exchange. Thus, the integration of the scheduling activity within the hierarchical planning pyramid is one of the driving forces of this contribution.

A central component in the validation and verification process of any scheduling approach is the set of computational experiments employed to evaluate it from various points of view, like the ones of solution quality, computational requirements,

robustness, scalability, extensibility, usability, flexibility, etc. Among the reasons for not making a comprehensive testing of the various scheduling formulations developed up to date, the lack of an appropriate problem library should be mentioned. However, not only a library of case studies is required, but also a way to specify industrial-relevant case-studies in a formal, machine understandable way. As shown is Section 4, current example descriptions combine textual and informal graphical representations (such as the STN or RTN graphs, precedence-based blocks, etc.), that may also have some textual annotations [1]. These descriptions might be vague, have a limited expressive power, and may lead to ambiguous interpretations. In some cases, supplementary material containing example data is provided [9], but the format varies from one contribution to the other and it is generally linked to both the notation and the mathematical programming software that has been adopted. Consequently, there is a need for a more formal specification of problems that can avoid these difficulties. Such a specification can then be employed to automate the generation of models and data required for the various approaches, thus improving the usability and spread of usage of the scheduling tools; in addition, comparisons can be promoted. Therefore, these new concerns constitute the second driving force of this contribution.

3   SchedOnto: Ontological Engineering Approach

Ontologies are semantic structures encoding concepts, relations and axioms, which provide a conceptual model of a given domain. Their aim is to capture consensual knowledge in a generic way that may be reused and shared across software applications and by groups of people [10-11]. Ontologies are widely employed for distinct purposes by different communities, but in the last decade they became popular when they turned into the backbone of the Semantic Web. In addition, ontologies are nowadays also setting the grounds for the integration of software applications.

An extensive state-of-the-art overview of methodologies for ontology engineering has been reported in [10]. This review points out different principles, design criteria and stages for ontology development. However, all of them involve at least the three stages proposed by the Enterprise ontology methodology [12] to build an ontology from scratch: (i) to identify the purpose and scope, (ii) to capture the concepts and relationships among these concepts, as well as the terms used to refer to concepts and relationships, and (iii) to codify the ontology.

3.1   Methodological Considerations

For the development of SchedOnto, and ad-hoc methodology based on well accepted principles has been proposed. It has the following four stages:
— Requirements specification; this stage identifies the scope and purpose of the ontology.
— Conceptualization stage, which organizes and converts an informally perceived view of the domain into a semi-formal specification using UML diagrams.

— Implementation stage, which implies the codification of the ontology using a formal language.
— Evaluation stage, which allows making a technical judgment of the ontology quality and usefulness with respect to the requirements specification, competency questions and/or the real world.

It should be noted that these stages are not truly sequential; indeed, any ontology development is an iterative and incremental process. If some need/weakness is detected during the execution of a stage, it is possible to return to any of the previous ones to make modifications and/or refinements. The two first stages have been completed and the last two are currently in progress. Some highlights of these methodological steps are given in the remaining of this section.

Requirements specification. This first step involved an analysis of needs and demands for different types of batch processes, plant environments, and operations modes, when addressing a scheduling problem, along with a comprehensive bibliographical research. Based on the gathered knowledge it has been possible to recognize the following modeling requirements. The goal has been to provide a formal specification of the scheduling domain by identifying relevant objects and relationships that:
— Represent input information necessary for the scheduling activity: (i) products and their master recipes, which specify how to manufacture them in a given site, (ii) production environment characteristics (equipment features, plant topology, etc.), (iii) production requests (manufacturing orders/amounts to be produced/demands for various products, due dates, etc.), (iv) resource (personnel, utilities, raw materials, equipment, etc.) availability along the scheduling horizon, and any other pertinent data.
— Explicit capture of the outcomes of the scheduling function: (i) control recipes that reflect how each batch is going to be produced (instantiations of master recipes), (ii) schedule specific information, detailing the agenda of each resource, etc.
— Allow the representation of production execution information, including timely data of how things have progressed, as well as batch-specific history information that could be used for rescheduling activities and managerial purposes, like performance analysis.

Conceptualization stage. The second main step in SchedOnto's development process required identifying and capturing the domain concepts and their relationships, trying to fulfill the previous requirements. To support this activity, UML (Unified Modelling Language) [13] was adopted. In addition to class diagrams, constraints about the objects in the model and invariants on classes have been added using OCL (Object Constraint Language) [14]. The results of this stage are not described due to lack of space. However, a partial model will be described in the next section.

Since ontologies are, by definition, based on consensual knowledge, both the ISA-88 [15-18] and ISA-95 [19-20] standards, which are well accepted in the industrial domain, have been taken into account during the conceptualization step. The mail goal of ISA-88 is the control of the batch process, whereas the final goal of the ISA-95 standard is the exchange of information between levels 4 and 3 of its hierarchical

model. Though both have a close relationship with the scheduling activity, they differ in terms of their purpose and terminology. According to the ISA-95 standard, the scheduling functions that are of interest (i.e. determination of production schedule, raw material requirements identification, etc.) interface to the manufacturing operations and control system ones through product definition information, production capability information, production schedule, and production performance information. A detailed analysis of this standard shows some overlapping with the information and activities handled by the ISA-88 one (e.g. product definition vs. recipe specification, equipment capability vs. physical model, etc.), which reveals some possible collision points. These issues, which have already raised some concerns [21-22], have been taken into account in the conceptualization stage.

In fact, the different parts of both the ISA-88 and ISA-95 standards have been developed by different people and at distinct moments. This led to the presence of inconsistent and/or incomplete information. Thus, not only there are some consistency problems that still need to be addressed when using these two standards together, as the scheduling domain requires, but also there might be some incoherencies within each standard itself. For instance, Part I of the ISA-88 standard [15] was approved in 2010 and Part II [16] more than ten years ago. This brings about some lack of correspondence between the terms and definitions included in Section 3 of [15] and the data structures that are specified in Part 2 of the same standard [16]. These matters have also been carefully considered during the ontology development process.

Implementation stage. The following planned activity in the methodology required the selection a formal language suitable for the codification of the concepts that were identified in the previous stage. Based on its ample acceptance, OWL 2 [23], developed by the W3C (World Wide Web Consortium), was chosen.

Evaluation stage. The development of SchedOnto has been guided by the principles of coherence, conciseness, intelligibility, adaptability, minimal ontological commitment and efficiency. Some of these principles are conflicting among themselves. Due to such incompatibilities, a suitable balance between the clashing principles was sought. Nowadays it is widely accepted that there is a lack of a formal methodology that considers all these criteria, which could be applied to evaluate domain ontologies. According to some authors [10], the ontology evaluation phase comprises three aspects: (i) ontology validation, (ii) ontology verification, and (iii) ontology assessment. Validation and verification activities are associated with a technical judgment of the content of the ontology with respect to a frame of reference, which can be requirement specifications, competency questions, or the real world. In turn, assessment focuses on judging the ontology content from the user's point of view. To address this issue it is expected to employ SchedOnto in the development of different types of applications, in distinct contexts. As mentioned in Section 2, these applications will have as targets: (i) integration issues, (ii) scheduling problems specification, (iii) supporting a scheduling benchmarking framework. Up to now, SchedOnto has been successfully employed to develop a reactive scheduling framework based on a hybrid approach that relies on an explicit domain representation and a CP model [24].

With respect to the integration of heterogeneous information sources, a set of ontologies will be used to provide the semantics enabling the data mediation. For each application/data source to integrate a local ontology will be defined in order to make explicit the knowledge implicitly encoded in it. In addition, SchedOnto will serve as a reference ontology that unifies the terminology and conceptualization of the domain. The ontologies will be part of a platform having a three-layer architecture: a) the reference one, containing SchedOnto, b) the application/data sources to integrate, and c) the intermediate or connector layer, which contains a connector component for each application/data source (A/DS) to be integrated. These connectors, based on local ontologies, will be in charge of coping with the heterogeneity issues posed by each A/DS, which implies, among other things, translating the queries/information needs expressed in terms of local data sources into ontological instances stated in terms of the reference ontology, and vice versa. The connector mediation process will be carried out in two steps, one syntactic and one semantic. For instance, for lifting a case study represented in one of the popular formats (e.g., the State-Task-Network based representation used by an MILP model) it will be required to perform a syntactic transformation in which the XML representation of the example is translated into instances in terms of a local ontology; then, in the semantic step these local ontology instances are transformed into instances of the reference ontology. Likewise, in order to lower a SchedOnto compliant case study to the format employed by a given MILP solution approach, the same steps will be applied in the opposite order.

Another research line, regarding ontology quality evaluation, analyses the structural dimension and employs an estimation of the ontology complexity considering the depth of the class hierarchy, as well as the number of classes, relations and instances defined in the ontology. This structural evaluation is under way.

4 Formal versus informal temporal representations

One of the weakest points of the ISA-88 standard is the representation of logical and temporal issues, which are generally dealt by means of textual annotations and graphs, like the Procedure Function Chart (PFC). These representations are not formal, can be ambiguous, and cannot be interpreted by a computer. For instance, let's consider the `Building Block`, which is one of the elements of a `Recipe`. In Table 5 of [16] it appears that one of its attributes is `UsageConstraint`, which has the following definition: "Defines other rules that determine the usage (e.g., "always succeeded by..." or "never runs in parallel with...")". Another illustrative example is the PFC, which depicts procedural logic to define the execution sequence of the procedural elements that comprise a recipe. It is described in detail in chapter 6 of [16]. It is easy to recognize that the logical aspects (alternative versus simultaneous sequences of recipe procedural elements) of this graphical representation, as well as its temporal and synchronization issues, are modeled by means of different types of vertical and horizontal lines, and by interpreting for the relative position of the procedural elements along the y-axis.

On the other hand, the temporal relations that SchedOnto requires, appearing in recipes, schedules, etc., have been explicitly modeled as shown in Fig. 1. It presents a partial view of the adopted temporal model, which is based on Allen's temporal relations [25]. This model is complemented by formal specifications of constraints.
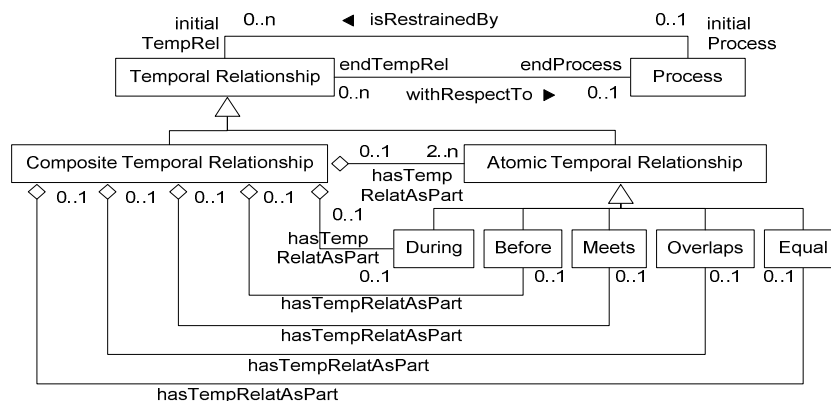

Fig. 1. Partial view of the temporal model adopted in SchedOnto

5   Conclusions

The paper described some features and the development process of SchedOnto, a domain ontology, which captures information of the scheduling field. Ontologies like SchedOnto play an essential role in describing and understanding complex fields. As a shared notation and a conceptual foundation, it might facilitate the communication, discussion, exchange of case studies, etc., among the members of scheduling community. In addition, since knowledge is explicitly and formally expressed, it supports inference processes and, therefore, the development of intelligent systems [24]. Last but not least, by providing a declarative, machine readable representation, SchedOnto can enable unambiguous communication between software agents that would otherwise be difficult or impossible. In this way, it can play a central role in solving nowadays integration problems that appear in the enterprise hierarchical planning pyramid. To illustrate the benefits of having a formal representation, the modeling of temporal aspects that are needed to represent recipe procedures was described and was contrasted with the graphical representation of PFCs proposed in [16].

References

1. Maravelias, C.: General Framework and Modeling Approach Classification for Chemical Production Scheduling, AIChE Journal, 58, 1812–1828 (2012)

2. Henning, G. P.: Production Scheduling in the Process Industries: Current trends, emerging challenges and opportunities, In: R. M. de Brito Alves, C. A. Oller do Nascimento and E. Chalbaud Biscaia, Jr. (eds), Computer-Aided Chemical Engineering, vol 27, pp. 23-28, Elsevier Science Ltd, United Kingdom (2009)

3. PABADIS' PROMISE Ontology. http://www.uni-magdeburg.de/iaf/cvs/pabadispromise/dokumente/Del_3_1_Final.pdf

4. Shobrys, D.E, White, D.C.: Planning, scheduling and control systems: Why cannot they work together? Computers and Chemical Engineering, 26 (2), 149-160 (2002)

5. Maravelias, C., Sung C.: Integration of production planning and scheduling: Overview, challenges and opportunities, Computers and Chemical Engineering, 33, 1919-1930, (2009)

6. Kreipl, S., Dickersback, J.T., Pinedo, M.: Coordination Issues in Supply Chain Planning and Scheduling. In: Herrmann, J. (ed), Handbook of Production Scheduling, pp. 177-212. Springer (2006)

7. Harjunkoski, I., Nyström, R., Horch, A.: Integration of Scheduling and Control – Theory or Practice? Computers and Chemical Engineering, 33, 1909-1918 (2009)

8. Muñoz, E., Espuña, A., Puigjaner, L.: Towards an ontological infrastructure for chemical batch process management, Computers and Chemical Engineering, 34 668-682 (2010)

9. Castro, P., Harjunkoski I., Grossmann, I.: Greedy Algorithm for Scheduling Batch Plants with Sequence-Dependent Changeovers, AIChE Journal, 57, 373-387 (2011)

10. Gómez-Pérez, A., Fernández-López, M., Corcho, O.:Ontological Engineering: With Examples from the Areas of Knowledge Management, E-Commerce and the Semantic Web, 2nd edition. Springer (2004)

11. Studer, R., Benjamins, V.R., Fensel, D.: Knowledge Engineering: Principles and Methods. IEEE Transactions on Data and Knowledge Engineering, 25, 161-197 (1998)

12. Uschold, M., King, M., Moralee, S., Zorgios, Y.: The Enterprise Ontology. Knowledge Engineering Review, 13, 31–89 (1998)

13. Object Management Group. UML Unified Modeling Language Specification V2.1.1: Superstructure, OMG, `http://www.omg.org`

14. Object Management Group. UML 2.0 OCL Specification, `http://www.omg.org`

15. ANSI/ISA–88.00.01: Batch Control Part 1: Models and Terminology (2010)

16. ANSI/ISA-88.00.02: Batch Control Part 2: Data Structures and Guidelines for Languages (2001)

17. ANSI/ISA-88.00.03: Batch Control Part 3: General and Site Recipe Models and Representation (2003)

18. ANSI/ISA-88.00.04: Batch Control Part 4: Batch Production Records (2006)

19. ANSI/ISA-95.00.01-2000: Enterprise-Control System Integration. Part 1: Models and terminology. ISBN/ID: 1-55617-727-5 (see also IEC 62264) (2000)

20. ANSI/ISA-95.00.03-2005: Enterprise-Control System Integration. Part 3: Activity models of manufacturing operations management, ISBN: 1-55617-955-3 (2005)

21. Scholten, B.: Integrating ISA-88 and ISA-95, ISA EXPO 2007, Houston, TX (2007)

22. ISA-TR-88.95.0.: Using ISA-88 and ISA-95 Together (2008)

23. W3C OWL Working Group, OWL 2 Web Ontology Language Document Overview. Technical Report. `http://www.w3.org/TR/owl2-overview/`

24. Novas, J.M., Henning, G.P: A Reactive Scheduling Framework Based-on Domain Knowledge and Constraint Programming, Computers and Chemical Engineering, 34, 2129-2148 (2010)

25. Allen, J.: Maintaining Knowledge about Temporal Intervals. Communications of the ACM, 26, 832-843 (1983)