ECP01 Workshop on Planning and Scheduling Technologies in New Methods of Electronic Mobile and Collaborative Work, 2001.

Planning-based generation of process models

MD R-Moreno[1], D. Borrajo[2], D. Meziat[1]

[1] Departamento de Automática. Universidad de Alcalá. Ctra Madrid-Barcelona, Km. 33,600
28871 Alcalá de Henares (Madrid), Spain.
{mdolores, meziat}@aut.uah.es
[2] Departamento de Informática. Universidad Carlos III, Avda. de la Universidad, 30
28911 Leganés (Madrid), Spain
dborrajo@ia.uc3m.es

Abstract. One popular way to model how organisations work is through their internal processes. In the case of current economy, especially in the case of organisations doing business in Internet, the definition of those processes has the problem of continuous adaptation to market needs. Therefore, organisations need powerful tools for automatic modelling, simulating and optimising those processes in such a way that the generated models comply with business rules. This is what the field of Business Process Reengineering (BPR) tries to solve. From an Artificial Intelligence (AI) perspective, planning and scheduling tools can be applied for this task, since they provide a declarative representation of the knowledge on the activities of processes, as well as means to generate only valid process models.

However, one of the main obstacles in applying AI planning techniques to real problems is the difficulty to model the domains. Usually, this requires that the people that have developed the planning system carry out the modelling phase since the representation depends very much on a deep knowledge of the internal working of the planning tools. Since, in BPR, there has already been some work on the definition of languages that allow non-experts to enter knowledge on processes into the tools, we propose here the use of one of such languages to enter knowledge on the organisation processes. Once the BPR expert has defined the domain using this language, we use an interface to translate such knowledge for input to an AI planning tool. This planner automatically generates the corresponding process models that are translated back to the process modelling tool.

1 Introduction

The use of the Web for carrying out business requires, among other issues, the restructuring of organisations' processes. This is specially needed for all those processes that deal with the relations with the customer (B2C), or with other organisations (B2B). Currently, there are computational tools for executing (enacting) and monitoring those processes. These tools are called workflow tools. Process

models (as well as information models) have to be given as an input to those tools. Under this context, any organisation can be seen as a set of agents executing processes (in parallel or in sequence) where each process is composed of a set of activities linked by dependencies [9].

Usually, the task of defining those models is performed with the aid of a set of tools that allow the graphical representation of them, together with the relations among the activities that occur within the processes. The task of defining the process models is performed by a human, who is an expert in the processes that take place in the organisation.

This situation needs to change in the Web context due to some reasons:

- Organisations that operate in the Internet have to be continuously adapting their processes to new issues that emerge literally everyday. Therefore, a re-designing of their processes has to be done periodically, requiring the continuous effort from someone of the organisation.
- Process models generated manually have to be checked for consistency and validity once they have been generated. Current process models authoring tools incorporate syntactic validity of the generated models. However, they do not have the ability to represent more semantic information about the processes, and, therefore, are incapable of checking for their semantic validity.
- Current process authoring tools only consider time and economic cost as quality measures. However, it is becoming now a need for organisations, especially within Internet-based business, to consider other quality-based aspects such as customer fidelity or quality of service.

Therefore, organisations need computational tools for automatically generating those process models that are able to:

- Generate process models from business specifications.
- Validate those models against rules of the organisation, as well as obtained patterns of its customers/partner organisations.
- Allow the representation of semantic oriented information, and not only syntactic information about activities of processes. At the same time, they should provide tools for handling the semantic information to obtain richer knowledge about the organisation behaviour.
- Allow the representation and reasoning about other quality criteria than just time and cost.

The field of AI planning and scheduling is a useful framework for providing reasonable solutions to those requirements [12, 14]. In this paper, we will use a rule-based tool, SHAMASH for the representation of semantic information of organisations' processes [16]. The represented knowledge is translated into the language of a planner, PRODIGY4.0, that is able to automatically generate the process models (plans in AI planning terminology) [20]. Then, those process models are translated back into SHAMASH, which can: semantically simulate the process; optimise the process; generate a process model able to be handled by a workflow tool;

and publish the process in HTML so that it can be accessed through the Internet by the allowed people (inside or outside the organisation). The advantages of this approach not only include the automatic generation of the models but also the detection of inconsistencies among tasks and a more domain dependent descriptive language for the planner.

The paper is structured as follows: section two introduces the definition of process models. Section three describes SHAMASH components and capabilities. Next, details of the integration of a common AI classical planner used to automate the modelling phase is described. Afterwards, we explain the integration with an example. Finally conclusions, future research and related work are presented.

2 Process modelling in organisations

A process model in an organisation can be seen as an abstract representation of the organisation elements and their behavioural, structural, informational and functional relationships. It presents a simplified view of the real world complexity. The organisation behaviour concerns the order in which the tasks to be performed are going to be executed, that is, the control flow. The organisation functionality refers to the functions the processes contain when the activities are being executed. The information aspect refers to the input data and the produced data. And the structure aspect relates to the organisation model (who is responsible of each activity) and its standards. The main concepts to be represented are [1]:

- tasks or activities that have to be done;
- agents (software or human) in charge of the tasks completion that belong to an organisational hierarchy;
- goals of the processes;
- data or information that will be transformed during the process; and
- business constrains.

To represent this knowledge, methods for process design as the IDEF Family [11] and UML [18] have been translated into automating tools that support them. The advantage of these methods is their popularity, so users are familiarised with its management. However, these methods do not communicate business processes effectively from an organisational perspective nor for workflow efficiency among organisations, and do not address responsibility and authority. Other more sophisticated tools allow a limited interaction with the users: the knowledge acquisition aspect is low but the description operation language is rich; that is, the syntax primes over the semantics.

In the middle of these approaches lie the Mixed-Initiative approaches [19]. These tools are designed to work with people, rather than completely taking over processing. They provide help on the interaction with the user, such that knowledge acquisition is integrated with problem solving. This allows the user to augment and adapt a system's knowledge in response to new situations and needs.

Most of the modelling tools allow a graphical representation of their processes. Some of them also simulate the models created but very few tools produce automatically and optimally the model bearing in mind the different organisation elements. Also a common criticism is that they do not produce output in a format that is acceptable to process enactment tools (Workflow tools) [1].

In the last few years, there has been considerable interest in the application of AI techniques to Workflow Management Systems to provide good modelling and reasoning solutions, enactment and monitoring of their processes and recovering techniques in case of failure [3,7,8,10]. If we concentrate on the reasoning aspect, AI planning tools can greatly improve the way in which organisation process models are generated. For instance, the AI community has addressed successfully the planning and scheduling issues in complex domains as military or space problems. Some researchers have seen the advantages of the integration of these fields, as it is shown by the existence of a Technical Co-ordination Unit of the European research network on planning and scheduling, PLANET[1], on applications of planning and scheduling to workflow. This has lead to some exploratory work reflected in a roadmap and some published papers [12,14]. However, to date few tools have been developed using these ideas.

As the result of a European funded project, SHAMASH [16], we have a software tool that is able to model realistically any organisation in terms of its standards, its resources (human or material), its structure (hierarchical or functional), and its processes. We refer to realistically because the user can represent the knowledge about not only the relevant concepts, but also the behaviour of these concepts. Given a process and an organisation model generated by the user, the tool is able to simulate the behaviour of the process and automatically generate better models. One of the bottlenecks of this process relates to the generation of the process models by the user. We propose the use of AI planning for this task.

3 SHAMASH. Knowledge Based Modelling of Processes

SHAMASH has been created as a Knowledge Based System using an object-oriented and rule-based approach with a modified RETE net algorithm as the inference engine. SHAMASH architecture is composed of four subsystems:

- Author subsystem: is in charge of the interaction with the user and allows to introduce the knowledge on the organisation into the system.
- Simulation and optimisation subsystem: performs an optimisation and simulation of the models created in the previous subsystem. While simulating a model we can observe the dynamic behaviour of the model. The user can analyse specific problems and measure the quality of the created model. Then if the behaviour is not as good as expected, the user can use the optimiser to obtain alternative models that improve the behaviour. Optimising a model consists on detecting problems, like bottlenecks, idle resources, or an abuse of agents. The optimiser looks for

[1] http://planet.dfki.de/

alternative models that solve this aspect and evaluates the quality indicators given by the simulation process for this new model. For the alternative models the optimiser uses a set of expert heuristics and alternative designs. All the new models are simulated and validated again.

- Text generation subsystem: produces an HTML version of the process model describing both standards and processes diagrams coherent with the graphical process diagrams. Any modifications in the graphical design will imply changes in the generated text. This is a key issue for modern organisations, in order for their internal (and possibly external) people to access information about the processes with which they are interacting. This allows a better and faster understanding of why they are doing a given task, the precedence among tasks, or issues such as responsibilities for the completion of tasks.
- Workflow interface subsystem: translates the process models into the input to a workflow engine following as much as possible the Workflow Management Coalition standard (WfMC) [21]: Workflow Process Description Language (WPDL).

3.1 SHAMASH as a Knowledge Based System

The Knowledge Base of SHAMASH and the applications developed with it contain all the knowledge about the domain: the objects, their methods, and rules to define the processes and validation behaviour. The object-orientation helps in two ways: first, it is a good way to structure the knowledge on processes and organisations; second, it can be user extendible by using object inheritance.

When defining a domain, the user starts by selecting and adding, through the interface, the new objects and attributes that belong to the process. In SHAMASH the main classes to be represented from the organisation are:

- CActivity: allows to define the types of activities involved in the process. For instance, in a billing process, receiving an invoice, signing a document, or approving a given payment.
- CElement: allows to define the elements that flow through a process. That is, the inputs and outputs of the tasks. For instance, an invoice, or a payment document.
- CResource: allows to define the resources available in the organisation, such as people or software
- CRole: allows to define the roles that the people on the organisation can play in the mentioned activities, as Administrative or Department head
- CUnit: allows to define the units in an organisation that can be either individuals or groups of people, such as Administrative Department, or Mary Shefield

3.2 Rule Based Approach

SHAMASH follows a rule-based approach for defining the behaviour of some elements in the Knowledge Base. The main rationality for having used rules for defining behaviour was that the user companies selected them as the knowledge formalism that was closer for them to represent the knowledge about the behaviour of the elements of the processes. Rules can access any represented information about the organisation, its structure, its processes, as well as its business rules (or standards). Therefore, very rich semantic behaviour can be defined within the processes, which account for a more realistic representation of them.

In order to help the user on entering the rules, a syntax-driven Rule Editor was defined. In the Rule Editor, the user can easily define, modify or remove rules in order to describe the behaviour of standards and process, establish validation functions and optimisation criteria, etc.

For the integration of the deliberative planner PRODIGY with SHAMASH we had to translate from the BPR based language of SHAMASH into the PRODIGY language [20], as it is described below.

4 Automating the generation of process models

For a better understanding of the integration of SHAMASH-PRODIGY architectures, Table 1 outlines at a high level the concepts that AI planning & scheduling share with the Workflow community (for a more detailed description we refer to [12]):

Table 1 Concepts mapping between AI P&S and workflow.

AI P&S	Workflow
Planning and Scheduling	Modelling and Scheduling
Execution	Enactment
Operators	Activities, tasks, …
Initial State	Organisation, resources
Goals	Business goals, service provision, …

Once the user has defined all the activities and elements that could be part of the process, the task of connecting all the activities begins. This task can be performed by a user, or automatically generated using an AI deliberative planner. We have used the non-linear planner PRODIGY but other planners such O-Plan [6] or CNLP [13] could also have been used. The automatic process generation was sketched in [15] as follows: first, the system translates the SHAMASH objects and rules into the PRODIGY language. Then, PRODIGY produces an ordered sequence of activities. And later, this sequence is translated back into SHAMASH to be presented graphically to the user. This process is shown in Figure 1. The translation has in mind the different semantics of BPR and AI planning concepts as well as their similarity:

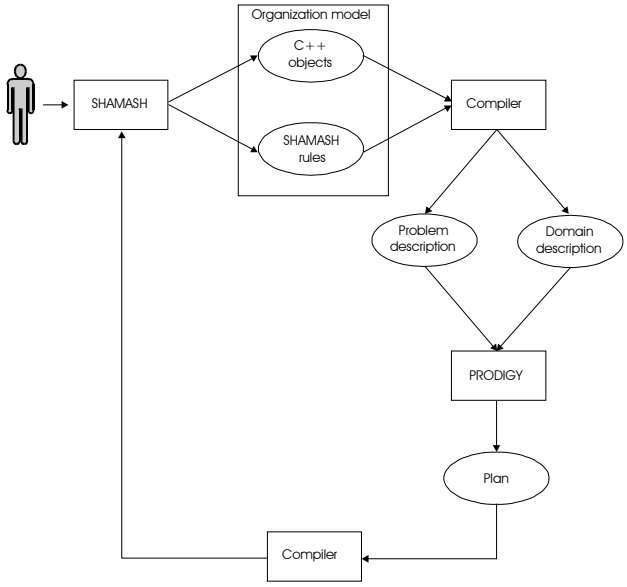

Figure 1. SHAMASH & PRODIGY integration.

Predicates: PRODIGY domain theory is an augmentation of the STRIPS representation. In this representation, a world state is represented by a set of grounded literals, the conjunction of which is intended to describe the given state. The states in SHAMASH are represented as C++ objects (that is, instances, attributes and their values). To represent them in PRODIGY we have generated, for each tuple <instance, attribute, value> in the SHAMASH KB, a binary predicate whose name is the attribute name. Its two arguments correspond to the identifier of the instance that it belongs to, and the value of the attribute. For instance, one of the attributes of the *order document* element in the billing process is the type of payment method (Visa, Check, Cash...). This is translated into the predicate:

```
(Payment <order document> Check)
```

Operators: task dependency is represented in BPR as activities with pre- and post-conditions (following the WfMC standard [21]). In SHAMASH pre-conditions are represented as rules that have to be fired before the activity can be executed. The activity post-conditions have to be true after the activity execution. On the other hand, PRODIGY has operators with pre-conditions and effects. Therefore, each activity name is translated into an operator name, the left-hand side of the pre-conditions of the activity into the operator pre-conditions and the right-hand side of the post-condition rules of the activity into the operator effects. Pre- and post-conditions refer to questions about the contents of the process KB at a given moment. Most of these questions refer to knowing the value of a given attribute of an instance, or setting it. In order to translate those questions and settings, the translation of attribute values into literals in the previous paragraph is taken into account.

Also, to connect activities in a process, they must have the same input/output elements. For instance, in the billing process (explained in more detail in the next section) the activity Bill precedes the activity Pay. To link them, the output element of the Bill activity (order document element) must be the same as the input element of the activity Pay. In PRODIGY input/output elements are operator parameters (params in Figure 3), that is, the variables that will appear with the operator name when it is instantiated.

Types: PRODIGY requires that each variable in the operators belongs to a user-defined type. Since SHAMASH follows an object-oriented approach, it represents all static knowledge as classes and instances. Therefore, every class in SHAMASH is translated into a PRODIGY type. Thus, the CElement will be translated as:

```
(ptype-of CElement: top-type)
```

and the classes that belong to the CElement (as the *order document* element) as:

```
(ptype-of COrder  CElement)
```

SHAMASH has also five basic types that are translated into PRODIGY user-defined types. For example, SHAMASH represents integer numbers as type integer. Given that PRODIGY has the capability of allowing to define variables with continuous values, called infinite types, this SHAMASH type is defined in PRODIGY as such.

Problem: apart from the domain theory, PRODIGY also needs the initial state and the goals to achieve. In SHAMASH the initial state is represented as the instances of the organisation units, roles, resources and elements defined by the user for each BPR domain (activities and organisation). For example, the instance that represents a human resource (e.g. Mary) can have its availability as one of its attributes. If it is considered a boolean attribute, this information is translated as:

```
(Availability Mary TRUE)
```

In relation to PRODIGY goals, we translate the user goals defined in the SHAMASH process into the PRODIGY problem goals. As an example, in a *Billing* process, the aim of the process might be defined as client satisfaction. If we define it as when the client receives the product, it could be represented in PRODIGY as:

```
(Product_Sent order_Client123 TRUE)
```

Plan: the planner generates a sequence of instantiated operators (activities). This sequence or plan represents the activities instances or tasks dependency in a process model. A plan in PRODIGY for a simplified process would look like:

```
(Receive_Client_Order order_Client123)
(Bill order_Client123 Mary_Shefield)
(Pay order_Client123 Check)
```

Then, the plan is translated back into SHAMASH. Once the activities are represented and linked in the interface, the user can change them if the results are not as expected or improve the model by simulating and optimising it.

5 An Example: An Internet Billing Process

Let us consider a simple scenario in order to clarify the concepts presented until now. The billing process could be considered as a process that every company has to face. The following is a simplified description of it. Suppose a client contacts the company (for instance, via the Internet) to buy some product. We identify the element o*rder* with the attributes: *ClientName* (type string), *Product_Number* (type integer), *Payment* (type string with two values: ”Check” or “Visa”), *Paid* (type boolean) and *Product_Sent* (type boolean).

Once the client has specified the products that s/he needs, the business process starts. A delivery note is generated with the cost of the products, and, at the same time, the goods are sent to the customer. If the customer has received the goods, it is time to pay by one of the payment types specified in the order. In this process three are the activities to perform: input client data (we call the activity Receive_Client_Order), generates the bill (Bill) and the payment of the products (Pay). The order document element is the input/output element of the three activities. As an example of a possible behaviour of the Bill activity, the rule describing its behaviour is shown in Figure 2. Figure 3 shows its corresponding PRODIGY operator translated by the system automatically from the rule.

```
Rule Bill with properties ruleset behaviour
 If Exists a COrder of type MetaClassElement named var
order1
    with Product_Number >( 0 )
    with Product_Sent = ( FALSE )
    with Payment  = ( “Visa” )
 Then Modify object order1
    with Product_Sent is  ( TRUE )
```

Figure 2. Example of a Bill activity rule in SHAMASH.

The pre-condition of the Billing activity checks if there is an element of the type class CElement (represented by the variable order1) with attribute values such that: the number of products that the client has bought (Product_Number) is greater than zero, the product has not yet been sent (Product_Sent) and the method chosen to pay (Payment) is Visa. As a post-condition, the execution of the activity will cause that the products bought are sent to the customer.

```
(OPERATOR  Bill
 (params <order1>)
 (preconds
   ((<order1> COrder)
    (<cct> (and integer
                (gen-from-pred
                    (Product_Number <order1> <cct>))
                (> <cct> 0))))
   (and (Product_Sent <order1> FALSE )
        (Payment <order1> Visa)))
 (effects
   ()
```

```
((add (Product_Sent <order1> TRUE))
 (del (Product_Sent <order1> FALSE)))))
```

Figure 3. PRODIGY operator corresponding to the Bill rule in Figure 2.

The variable *order1* is used to represent the element instantiated by PRODIGY among the *orders* that the problem might have (order_Client123, in the plan instantiated in the last section). The function *gen-from-pred* generates all the numeric values of the predicate *Product_Number* in the problem with a value greater than zero. This function is only used with infinite-type (numbers), as is the case of the integer type.

With respect to the problem definition, Figure 4 shows the initial conditions generated automatically from the organisation information. This includes issues such as who works in each unit (Assignedto Financial_Unit Mary), the cost per hour of the employees belonging to a given category (Cost Administrative 1230$) and other information related to the details of a specific order (in this example the order number 123). The Figure also shows the goals that PRODIGY needs to accomplish. In this case, the only goal is to send the products to the client whose order document is 123. With these conditions PRODIGY generates the plan described before, with resources, units and roles in that process.

```
(state
  (and
    (Availability Mary TRUE)
    (Cost  Administrative 1230$)
    (Id Financial_Unit  123A)
    (Assignedto Financial_Unit Mary)
    (ClientName order_Client123  SmithD.) ...
    (Product_Number order_Client123  2334)
    (Payment  order_Client123  Visa)
    (Paid order_Client123 YES)
    (Product_Sent order_Client123 FALSE)))
(goal
    (Product_Sent order_Client123 TRUE))
```

Figure 4. PRODIGY problem for the Billing example.

6 Related Work

Several research groups have integrated AI techniques with Workflow Management Systems (support the modelling and the enactment phase of the software engineering process). Some have applied planning tools in the enactment phase and very few have integrated planning techniques with BPR tools during the modelling phase.

The SWIM system [3] integrates process instantiation, task allocation, execution, monitoring and optimisation for improvement in the schedule. There is a Process

Library that provides the correct process definition to the Dynamic Process Manager. New processes can also be added to this library.

In [2] they use an improved version of the CNLP planner [13] that allows conditional temporal planning for concurrent execution. At modelling time there is an intelligent assistant to improve/complete the workflow model the user is working with.

The MILOS project [8] is a process modelling and enactment approach that provides project planning and management support like resource allocation and time scheduling for tasks in the project. The MILOS workflow engine allows the model and plan to be changed during project enactment and provides support for process restarts whenever necessary. The library of process models contains descriptions of best practices in software development for a given company. The project manager selects processes and methods from the library creating an initial project plan. The plan is uploaded to standard project management tools as MS-Project.

The TBPM project [9] is based on work carried out in the Enterprise project [17] and centres around an intelligent workflow engine that includes an interactive process planner. The planner uses AI techniques based on O-Plan [6] to assist in the tasks planning, while permitting the user to participate in planning decisions. An agent-based architecture supports the execution and co-ordination of the planned process among multiple participants and distributes processes across computer networks. The user is able to plan a task by assembling fragments and then to refine it, generating a hierarchical model of the process to be performed. For more flexibility, the user is able to edit the process model before and during its enactment, in order to specialise it.

In all these systems the modelling phase is based on a process library but there is not automatic generation as we have outlined in this integration of SHAMASH and PRODIGY. These systems do not exploit the advantages that a BPR tool as SHAMASH could offer to knowledge acquisition in AI planners.

7 Conclusions and Future Work

We have described the advantages of using an AI classical planner, PRODIGY, for modelling processes in SHAMASH, a Knowledge Based System that uses an object oriented and rule-based approach. The SHAMASH rules and objects are translated into PRODIGY types and operators to produce a plan that correspond to a process instance (tasks dependency). Each plan will vary depending on the resources available, elements that flow through the process and the different tasks that the organisations can introduce to adapt their processes to the market changes. With this approach the models generated are automatically validated avoiding inconsistencies in linking activities and saving time to the user.

We also wanted to study the issues that AI planners can gain with this approach. Generally, to specify the domain theory, a deep understanding of the way AI planners work and its terminology is needed. However, if we use a tool like SHAMASH, the description language is closer to the user and allows an automatic verification of the syntax through a friendly interface.

In this integration we could have also used the language for the AIPS-2002 Competition Committee (PDDL2.1) instead of PDL4.0 (PRODIGY's syntax). Then, any planner that supports PDDL2.1 could exploit the advantages of using a descriptive language as the one used in SHAMASH

Improvements to be addressed include using the B-PRODIGY planner [4] to build plans with conditional branches or the Analogy/CBR module [5] to reduce the search effort and reuse problem solving experience in terms of libraries of previously generated process models.

Acknowledgements

The SHAMASH project has being carried out in the course of the R&D project funded by the Esprit Program of the Commission of the European Communities as project number 25491. A complementary grant was given by the Spanish research commission, CICYT, under project number TIC98-1847-CE. We thank the partners of this project, who have originated and contributed to the ideas reported: UF (Unión Fenosa), SAGE (Software AG España), SEMA GROUP sae, UC3M (Universidad Carlos III de Madrid), WIP (Wirstchaft und infrastruktur & Co Planungs KG), and EDP (Electricidade de Portugal). We would specially like to thank all the UC3M team, the PLANET people and Paul Kearney. Through talks with him we have outlined many ideas. This work has also been partially funded by grant CICYT TAP1999-0535-C02-02.

References

1. Alonso G., Agrawal D., El Abbadi A. and Mohan C.. *Functionalities and Limitations of Current Workflow Systems*. IEEE Expert, 12(5), 1997.
2. Beckstein C. and Klausner J. *A planning framework management*. Sixteenth International Joint Conference on AI 1999.
3. Berry P.M., Drabble B. *SWIM: An AI-based System for Workflow Enabled Reactive Control*. In the 16th IJCAI99 Workshop on Intelligent Workflow and Process Management: The New Frontier for AI in Business. Mamdouth Ibrahim, Brian Drabble and Peter Jarvis editors.
4. Blythe J. and Veloso M. *Analogical replay for efficient conditional planning*. In Proceedings of AAAI-97, pages 668-673, Providence, Rhode Island, July 1997.
5. Borrajo D. and Veloso M. *Lazy incremental learning of control knowledge for efficiently obtaining quality plans*. AI Review Journal. Special Issue on Lazy Learning, Vol. 11 pages 371-405, 1997.
6. Currie K.W .and Tate, A. *O-Plan: the Open Planning Architecture.* Artificial Intelligence, Vol.51, Nº 1, North-Holland. Autumn 1991.
7. Goh A., Koh Y.-K., Domazet D.S. *ECA Rule-Based Support for Workflows.* Artificial Intelligence in Engineering 15 (2001) 37-46.
8. Goldmann S., Munich J., Holz H. *Distributed process planning support with MILOS.* In the Int. Journal of Software Engineering and Knowledge Engineering, October 2000.

9. Jarvis P., Stader J., Macintosh A., Moore J., and Chung P. *A Framework for Equipping Workflow Systems with Knowledge about Organisational Structure and Authority*. In Proceedings of the Workshop on Systems Modeling for Business Process Improvement (SMBPI-99), University of Ulster, County Antrim, Northern Ireland, UK, pp 205 - 219.

10. Jarvis P.., Stader J., Macintosh A., Moore J., Chung P. *What right do you have to do that?*. In ICEIS – 1st International Conference on Enterprise Information Systems. Portugal, 1999.

11. Mayer R. J., Painter M. and Witte P. de. *IDEF Family of Methods for Concurrent Engneering and Business Re-engineering Applications*. Knowledge Based Systems, Inc. 1992.

12. Paul Kearney and Daniel Borrajo. An {R\&D} Agenda for AI Planning applied to Workflow. Management Proceedings of the eBusiness and eWork Conference 2000.

13. Peot M.A and Smith D.E. *Conditional Nonlinear Planning*. In Proc. 1st International Conference AI Planning Systems, pp189-197. College Park, Maryland. Morgan Kaufmann.

14. PLANET Technical Coordination Unit on Workflow Management. http://www.labs.bt.com/projects/planet/

15. R-Moreno M.D, Borrajo D., Meziat D. *Process modelling and AI planning techniques: A new approach*. Second International Workshop on Information Integration and Web-based Applications & Services. IIWAS2000. Yogyakarta.

16. Sierra-Alonso A., Aler R., and Borrajo D. *Knowledge-Based Modelling of Processes.* In the 16th IJCAI99 Workshop on Intelligent Workflow and Process Management: The New Frontier for AI in Business. Mamdouth Ibrahim, Brian Drabble and Peter Jarvis editors.

17. Stader J. *Results of the Enterprise Project*. Proc. 16th Int. Conference of the British Computer Society Specialist Group on Expert Systems. Cambridge, UK, 1996.

18. UML notation guide. Rational software corporation. 1997. http: www.rational.com/uml

19. Valente A., Blythe J,. Gil Y., Swartout W. *On the role of Humans in Enterprise Control Systems: the Experience of INSPECT*. In the DARPA-JFACC Symposium on Advances in Enterprise Control, San Diego. November 1999.

20. Veloso M., Carbonell J., Perez A., Borrajo D., Fink E., and Blythe J. *Integrating planning and learning*: *The PRODIGY architecture.* Journal of Experimental and Theoretical AI, Vol 7, pages 81-120, 1995.

21. Workflow Management Coalition. http://www.wfmc.org/