10 PRODNET COORDINATION MODULE

L.M. Camarinha-Matos♦, C. P. Lima
New University of Lisbon, Portugal

The coordination model developed in PRODNET II for the virtual enterprise infrastructure is presented. The key issues in coordination are discussed, such as, flexibility and configurability, hierarchical models, events, synchronization, loops and cycles, and so on. The software modules involved in the coordination are described: the Local Configuration Module and the Local Coordination Module. Related works are discussed and some directions for further research are also given.

INTRODUCTION

The coordination functionality is a relevant part of a flexible infrastructure for virtual enterprises (VE). The infrastructure proposed by PRODNET II is intended to support a large diversity of enterprises and interconnection modes, varying from a small company with a networked PC, to a medium or large company with various legacy systems. As the infrastructure cannot impose tight rules to the companies, it must be configurable in order to support a variety of scenarios. For instance, in order to guarantee that both privacy and autonomy of each company are preserved within a VE environment, the behavior of the infrastructure must be *configured* according to the needs of each company. Different companies have different internal environments and they make business in different ways. For example, a company might require a strong human-based control of the clients' orders whilst others may prefer to rely on the PPC (Production Planning and Control) system. Therefore, it is necessary that to give each company the possibility of *explicitly configure its desired cooperation behavior*.

The PRODNET Cooperation Layer (PCL) provides the appropriate support for the interactions among companies in a VE environment. As PCL is composed by several modules that must work cooperatively as a single unit, a coordination / management function is required for these modules in order to guarantee their

♦ Corresponding author address: Universidade Nova de Lisboa, Quinta da Torre, 2825 Monte Caparica, Portugal, tel. +351-1-2948517, fax +351-1-2941253, e-mail: cam@uninova.pt.

The original version of this chapter was revised: The copyright line was incorrect. This has been corrected. The Erratum to this chapter is available at DOI: 10.1007/978-0-387-35577-1_37

L. M. Camarinha-Matos et al. (eds.), *Infrastructures for Virtual Enterprises*

harmonic operation and the achievement of the desired cooperation. Inside PCL, the *Local Coordinator Module* (LCM) is the component responsible for this coordination, whereas the *Local Configurator Module* (LCF) is the responsible for the configuration issues in PCL. AS configuration and coordination are inter-related issues in PRODNET II, LCM and LCF have to work in a close liaison. Figure 1 highlights these modules in the PCL architecture.

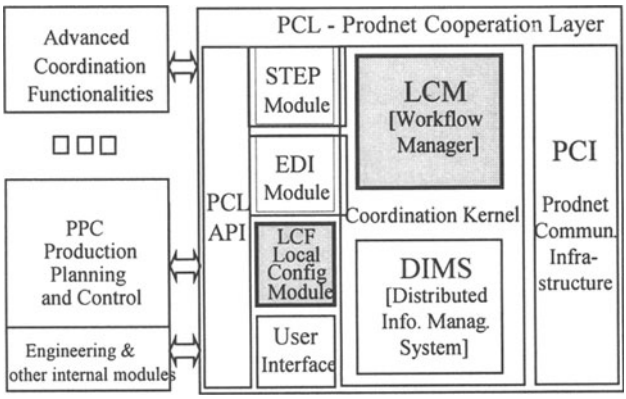

Figure 1 – Local coordination components in the PRODNET architecture

BASIC WORKFLOW CONCEPTS

Workflow Reference Model

In PRODNET II, like in some other research projects (Bakkeren et al., 1997), (Chan et al., 1998), (Alonso et al., 1999), strong similarities were identified between the VE configuration and management, and the functionalities that can be supported by a workflow system (Camarinha-Matos et al., 1997b). Therefore, the PRODNET II approach to implement the coordination structure inside PCL is based on the workflow reference model proposed by the Workflow Management Coalition (WfMC). This group is a non-profit, international organization whose mission is to promote the use of workflow through the establishment of standards for software terminology, interoperability and connectivity between workflow products (WfMC, 1994, 1998).

The WfMC reference model (Figure 2) identifies the major functional areas that are part of a workflow-based scenario. The *Process Definition* contains specifications for process definition data and data interchange with the workflow enactment service. The *Enactment Service* is the workflow executor itself, which can be implemented by one or several workflow engines. Both *Invoked and Client Applications* represent a variety of IT application types that support the execution of workflow models. *Administration and Monitoring* tools provide system monitoring and metric functions to facilitate the management of the workflow enactment service. Finally, the *Interoperability Interface* provides support for the interoperation between different enactment services.

The benefits to PRODNET II resulting from the adoption of this model are the following:

- The WfMC is a widely representative consortium in the area of workflow management systems, therefore using the reference model, a *de facto* standard, facilitates the integration of PRODNET and legacy systems;
- The model includes a formal language to support the workflow model definition – the Workflow Process Definition Language (WPDL) (WfMC, 1998). PRODNET uses WPDL as the workflow model representation format, which guarantees that PCL can import workflow models created by WPDL-compliant editors;
- The Workflow Engine is the executor of a workflow model. As the coordination mechanisms inside PCL adopted some concepts from this component, the LCM can be viewed as a specialized workflow engine executing workflow models customized for each company;
- The other PCL modules can be viewed as Invoked Applications, that provide services to support the LCM operation;
- When a company participates in various VEs, the multi-VE environment can be configured in a workflow model, by using one workflow definition per VE.

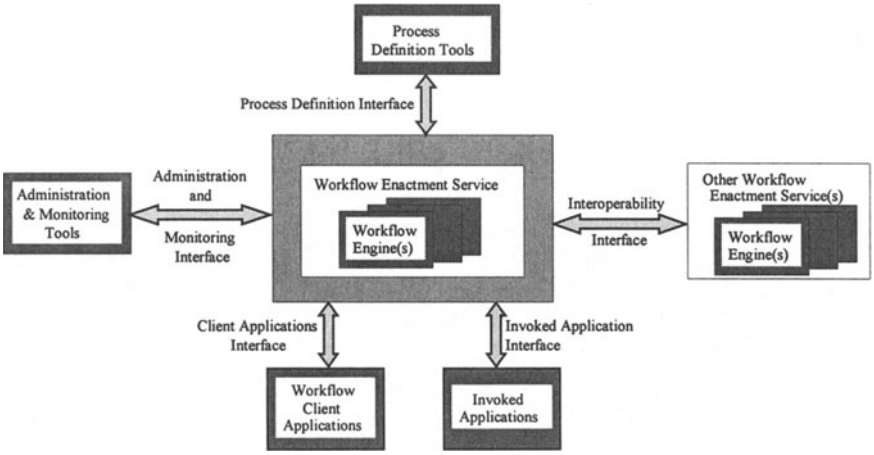

Figure 2 – WfMC Reference Model (WfMC,1994, OMG, 1998)

PCL COORDINATION MECHANISMS

The generic goals to be achieved by the PCL coordination mechanisms are: i) to support coordination of activities among PCL modules; ii) to be flexible and customizable by each company according to its particular needs and preserving its privacy and autonomy when interacting in a VE environment; and iii) to provide support for coordination among the VE members in order to guarantee the accomplishment of the common VE goal.

As the underlying infrastructure, a workflow management system is developed, comprising a graphical workflow editor, a workflow engine, and a workflow monitor. The workflow model or "plan" is a user-defined template to guide all actions executed by the workflow engine, which provides a strong flexibility and facilitate customization in terms of what has to be coordinated by PCL. A graphical language to support the workflow model definition is specified and the corresponding workflow editor is implemented. The editor checks the consistency of the model and generates a WPDL file. The workflow engine executes the actions specified in the workflow models. Various workflow models can be loaded into the working space of the workflow engine and turned active, i.e. launched in execution simultaneously. Specific events, such as the arrival of an EDI message, start the execution of specific workflow plans. The workflow monitor is used as an user front-end to monitor the status of execution of active workflow plans and to interact with the workflow engine.

PCL Services

Service is a key concept in PCL coordination. From the enterprise applications point of view, PCL is a "black box" offering a set of services. There are three types of services defined inside PCL to implement the cooperation functionalities of the enterprise: *Core*, *Auxiliary* and *Internal services*, as illustrated in Figures 3 and 4. *Core Services* are "macro services" offered by PCL to the enterprise applications, such as: PCL_SendingOrder, PCL_ReceivingCondraMessage, PCL_GetBPKey, etc. These services are defined / configured for each enterprise.

Each core service is configured as a workflow model and is composed by a sequence of activities connected by transitions. An activity can be implemented by invoking a PCL *auxiliary service* which means that an activity has to invoke some service(s) provided by other PCL modules, namely EDI, PCI, DIMS, or STEP. For instance, DIMS_AttachFilesReference is an auxiliary service used by LCM when sending out a CONDRA message. An *internal service* is a functions offered by one module that is directly invoked by other module without the intervention of LCM. For instance, DIMS_GetStepDataExchange is an internal service offered by DIMS that is used by the EDI module to read STEP data from the DIMS.

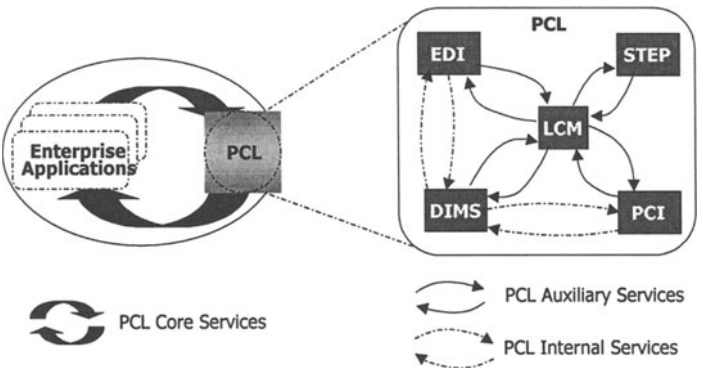

Figure 3 – PCL context

PCL Core Service

For each enterprise, several core services, i.e. several workflow plans need to be defined. These plans, in principle, define the execution steps for two categories of events:

1. Inter-enterprise events (i.e., requests arriving another enterprise), and
2. Intra-enterprise events (i.e., requests arriving either from the internal enterprise applications, or from PCL components).

The first category occurs, for instance, when it is necessary to process the sending of a purchasing order from enterprise A to enterprise B, or when it is necessary to exchange a product model between two companies, during a product negotiation process (supporting cooperative design. An example of the second category is a request for retrieving some data from the Distributed Information Management System (DIMS).

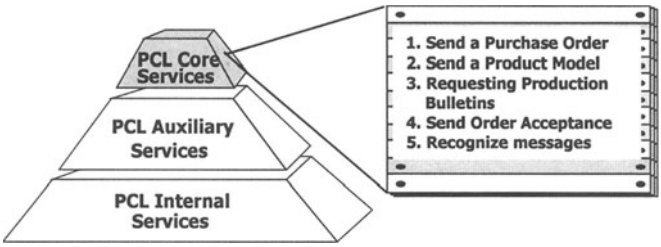

Figure 4 – Classes of PCL services and examples of Core Services

LCM uses the workflow models to handle events from both categories, based on a uniform mechanism. From the LCM perspective, the reaction to any event comprises a number of steps that have to be executed in the sequence configured by a human operator. Each sequence of steps is a *workflow plan* that implements a *PCL Core Service*. Each company, according to its internal rules, configures such services. PCL core services are mainly devoted to support the interoperation of the enterprise with its partners, in a VE environment. Figure 4 shows some examples of core services.

Each core service configured in a workflow model is therefore composed of a sequence of activities connected by transitions. An activity (box in Figure 5a) can be implemented by a sub-workflow or by a PCL auxiliary service. In other words, the execution of an activity invokes either a sub-workflow model or one of the services provided by other PCL modules, namely by the EDI, PCI, DIMS, or STEP modules. Examples of PCL auxiliary services EDI_BuildMessage, or PCI_DeliverMessage. A sub-workflow model can be seen as a particular task that frequently occurs in a certain company.

Core services might have input and/or output parameters. The input parameters have to be passed by the enterprise application (such as PPC, a PDM Editor, etc.), when a core service execution is invoked. The core services output parameters will be sent back to that application, when the core service execution is finished. The graphical symbols used to represent a workflow plan are shown in Figure 5b.

It is important to notice that the separation between core services configuration

(definition) and their enactment provide the same advantages of the separation of interfaces and code in modular and object-oriented programming (Alonso et al, 1999).

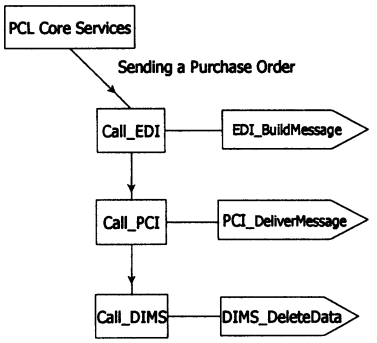

Figure 5a – Illustrative workflow plan for the PCL core service "Sending a Purchase Order"

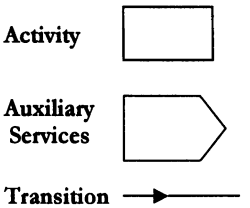

Figure 5b – Graphical notation

Typically, for different companies the procedure (process) of sending an order is different. Using the LCF editor it is possible and rather simple to edit different workflow plans representing different behaviors for sending an order. Clearly, another workflow plan has to be defined for the reception of a purchase order message at the receiving enterprise. Similarly, for every possible kind of event, it is necessary to create the corresponding workflow plan / core service.

Flexibility and Configurability

Configurability is a major issue in the PCL coordination mechanisms. Furthermore, the configuration process must be as flexible as possible in order to cope with the high level of heterogeneity found in a VE environment, especially considering that PRODNET II is strongly focused on SMEs.

Through the user-friendly graphical workflow editor developed in LCF, the PCL coordination model supports these features. Each company is able to configure the workflow models, in terms of:

i) The set of core services that compose each workflow model, i.e., what are the services offered to / requested from a VE. For instance, in one VE a company might want to send and receive production history bulletins, while in another this service is not relevant.

ii) PCL Core services input and output parameters: what is the information required supporting the execution of each core service. For instance, when sending a purchase order to a certain VE partner, it might be necessary to include a digital signature, which requires a specific parameter.

iii) The sequence of actions involved in the execution of a core service, i.e., specification of the PCL auxiliary services supporting the execution of such core service. For instance, the audit control might be required when exchanging orders with some VE partners.

iv) The dynamic data flow management to support each core service execution,

which is strongly related to point iii).

v) The definition of sub-workflow models to support repetitive tasks, as well as the construction of hierarchical workflow models.

Some of these concepts will be further detailed along this section.

Hierarchy and Reusability of Tasks

As mentioned above, in a workflow plan, an activity can be *atomic* or *complex*, based on its implementation. An atomic activity is implemented by invoking a single auxiliary service. A complex activity is defined recursively, implemented as *a sub-workflow*, which means that such activity, in fact, starts the execution of the new flow (the sub-workflow). Analogous to a workflow, a sub-workflow is also composed by activities that invoke auxiliary services and are connected by transitions. This mechanism supports the execution of a core service in different levels.

Besides the activation of the sub-workflow, another feature of the PCL coordination mechanisms is the ability to pass parameters to a sub-workflow. For illustrative purposes, Figure 6 shows a workflow model in which the *Activity 2* is implemented by the sub-workflow *ABC*, which is composed by two branches. When it is invoked, an input parameter might be passed in order to define which branch has to be executed. In the example, the valid values for the input parameter are "10" or "20".

Sub-workflows also provide a basic degree of *reusability* in the model, since a sub-workflow can be used several times in a workflow model. The more frequent tasks can be modeled as sub-workflows and used as many times as necessary. This feature allows the creation of a library of sub-workflows that model the tasks frequently performed.

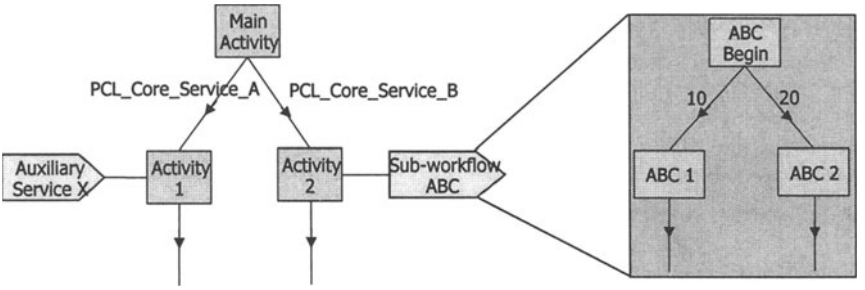

Figure 6 – Activity implemented as sub-workflow

Data flow management

PCL services and sub-workflows have input and output parameters. The input parameters are received when these services / plans are invoked. The output parameters are sent back to the "activator", when the execution is finished. Therefore, it is necessary to provide a data management support in order to cope with the passing of parameters among auxiliary services and sub-workflows.

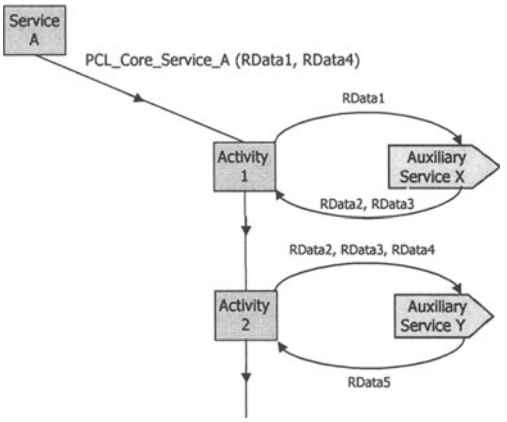

Figure 7 – Data management in the PCL coordination model

PCL uses the entity *Relevant Data,* from the WfMC model, to support the parameters. Each relevant data item represents one variable that can be used both as input and/or as output parameter. Parameter passing in the PCL coordination mechanisms can be totally configured by the human operator. The PCL user can create as many variables as necessary to handle the data sent to / received from the auxiliary services / sub-workflows. The valid types of these variables come from the definition of the PCL auxiliary services parameters type (such as *PciDeliveryConditions, PclResultConditon, DimsDataKey*, etc.), and the basic types – such as *int*eger, *string, real, boolean*, etc.

Figure 7 presents an example of this mechanism. The PCL_Core_Service_A receives two input parameters, *RData1* and *RData4*. *RData1* is passed to the auxiliary service *X* that, on its turn, sends back the output parameters *RData2* and *RData3*. Both of them plus *RData4* are used to activate the auxiliary service *Y*, which sends back *RData5*. Analogously to a C program, both auxiliary services and sub-workflows correspond to procedures and the relevant data are variables that can be used as parameters to send data to and receive data from these procedures. LCM creates a *working space* for each active workflow plan in order to support these data management.

Events

There are four types of events relevant to the PCL coordination mechanisms: elapsed time, arrival of answer from invoked service, human interaction, and arrival of external message.

The activities can be *temporized*, meaning that each activity can be time-conditioned by a specific relative time (e.g. wait for 5 hours from the start of the activity) or absolute period (e.g. each month, from the 1st to 7th) to start its execution. The waiting time is user-defined.

The second type of event is based on the communication process developed to support the interoperation among PCL modules, which is based on a client/server architecture between the coordinator and other PCL modules. Auxiliary services are

asynchronously called by the coordinator module that will receive, later, an answer related to this call. The LCM maintains a list of suspended activities waiting for answers to specific requests. Each request is univocally identified within PCL.

The third type is related to the events generated by a human operator for either data input or operational control. For instance, an operational control event occurs when the human operator decides to stop a core service execution. Or the human operator can input some data that is expected along the flow, such as the acceptance or rejection of an order sent by a partner.

Finally the last class refers to the messages that come from the VE partners. These messages are requests or answers of/to services. For instance, when company A sends a purchase order to company B, the later might send "an event" back to A (e.g., *order acceptance* message) confirming the commitment to process that order.

Synchronization and Conditional Flows

PCL uses three constructs from the WfMC reference model to support synchronization and conditional flows, namely: conditioned transitions, splits and joins.

One activity can be connected to **'n'** activities and some logical conditions related to a group of transitions can be defined through the **split** mechanism (Figure 8). Two logical operators are supported: **and** and **xor**, which means respectively that all transitions are started as soon as the activity to which they are connected finishes (executed in parallel), or only one transition is started based on a certain condition value.

On the "end side" of a transition, a ***join*** mechanism can be used to support the "arrival" of several transitions at an activity. Similarly to the **split** mechanism, the logical conditions **and/xor** are also defined, working in the same way. It is important to emphasize that the *join* can be used to synchronize actions, due to the fact that one activity that receives a ***join and*** is only executed after all its predecessors finish their execution.

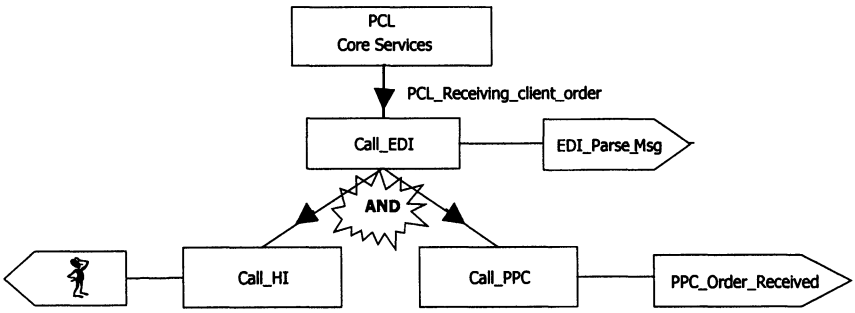

Figure 8 – ***Split and***: in this example both activities (Call_HI and Call_PPC) will be executed after the Call_EDI activity finishes

Each transition can have a specific condition associated to it, which allows the definition of conditioned flows. Each invoked service can return an "execution status result", which can be used to define a condition for the transitions that follow

the "invoking activity". For instance, in Figure 9, activity A11 follows A1 if service X (which implements A1) returns "10"; if the returned value is "20" the activity A12 starts. Any other value returned from service X will stop the flow execution and has to be treated by an exception handling mechanism.

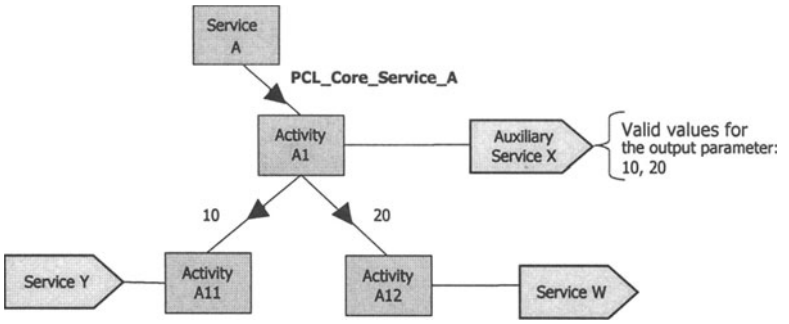

Figure 9 – Conditioned transitions

Cyclical and Periodical Tasks

The PCL coordination model also provides support for **cyclical** and **periodical** tasks. For instance, Figure 10 shows a workflow plan to support a case in which the service "Sending Purchase Order" has to be executed periodically, till the end of a particular contract. The first activity, "Wait for an Activation Date", is a temporized activity that waits for a specific time to start acting, which means, it precedes the next activity which sends an alert to the human operator, telling him/her that a purchase order has to be sent out. In alternative to defining an alert event, it is possible to define a service sending the purchase order automatically. However, it may be considered more reasonable that an event such as "sending a purchase order" is controlled by an operator. This loop remains valid as long as the contract that supports the business relationship is valid.

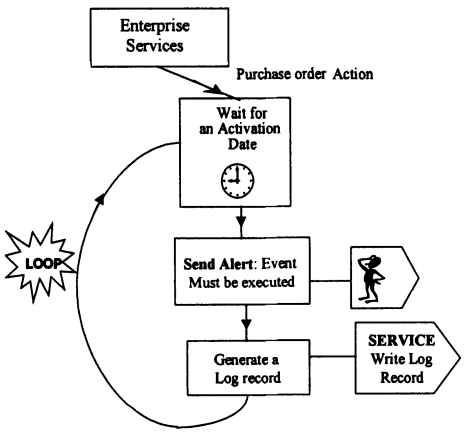

Figure 10 – Sending a cyclic Purchase Order alert

LOCAL CONFIGURATION MODULE

A key point in the PCL configuration is the ability to support a gradual migration of the company's legacy business practices towards a VE environment operation. In a first step, PCL has to guarantee that the company's privacy, autonomy as well as its way of making business will be preserved, as far as possible. When the level of trust between the enterprise and the VE partners increases, or a better understanding of the VE operation is achieved, a new behavior of the PCL should be defined. This is provided by the LCF module through a flexible configuration process, which is strongly based on an interaction with a human operator representing the company's needs and wishes.

The LCF deals with several classes of information (Camarinha-Matos at al., 1998b) for which various configuration modules are developed.

Graphical Workflow Editor

The graphical workflow editor is the component of the LCF that provides a user-friendly environment in which a human operator can create workflow models representing the desired behavior of the PCL when a core service is requested. As already mentioned, such workflow models specify the PCL coordination mechanisms. The editor uses a graphical language (Figure 5b) to represent the entities that compose the various core services. In a workflow model, each branch corresponds to one PCL core service. Figure 11 shows an example of workflow edition.

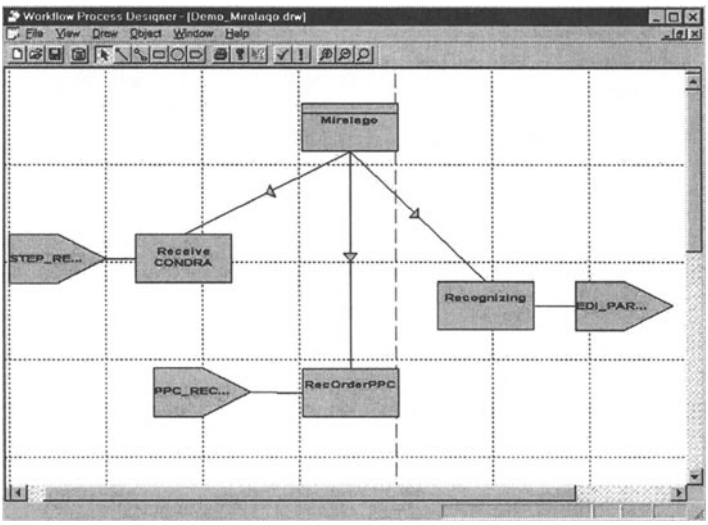

Figure 11 – Graphical Workflow Editor

To be loaded by the LCM module, a workflow model has to be represented in

the WPDL format. However, as the model has to follow some syntactic and semantic rules, the Editor provides a validation process. The errors are detected and the user receives warning messages identifying them. A partial example of a workflow model written in "WPDL" format is shown in Figure 12.

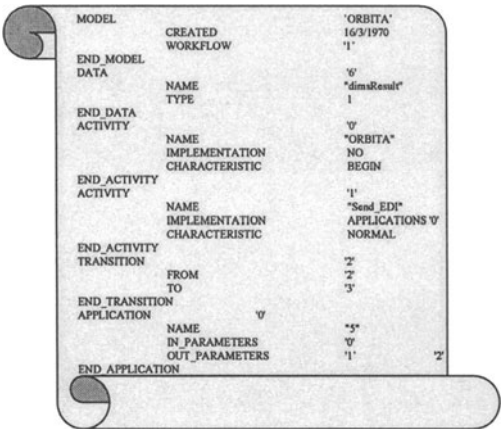

Figure 12 – Workflow model written in WPDL

Other Classes of Information

Other PCL modules have particular configuration needs, such as:

- The EDI module requires the configuration of the EDI subsets to be used between two VE members;
- The PCI requires a configuration of some issues related to the network itself and safety parameters;
- The DIMS module requires the definition of data access rights and visibility levels.

For this purpose, other specific modules have been included in LCF.

LOCAL COORDINATION MODULE

Two modules compose the LCM: the *Kernel* and the *Monitor*. The Kernel is a workflow engine and the Monitor provides both visualization of workflow execution and interaction with the Kernel (Figure 13).

The Kernel is composed of three major modules: the Loader/Parser, the Workflow Engine itself, and the Message Handler. The first one is responsible for loading and evaluating the "WPDL" models created with the workflow editor, and for preparing the data to be used by the engine.

The second module is the workflow executor, i.e. the real coordinator inside the Kernel. It has to keep track of the execution of core services and to coordinate the sending / receiving of messages along the model execution. The Message Handler is responsible for sending / receiving messages to / from other PCL modules.

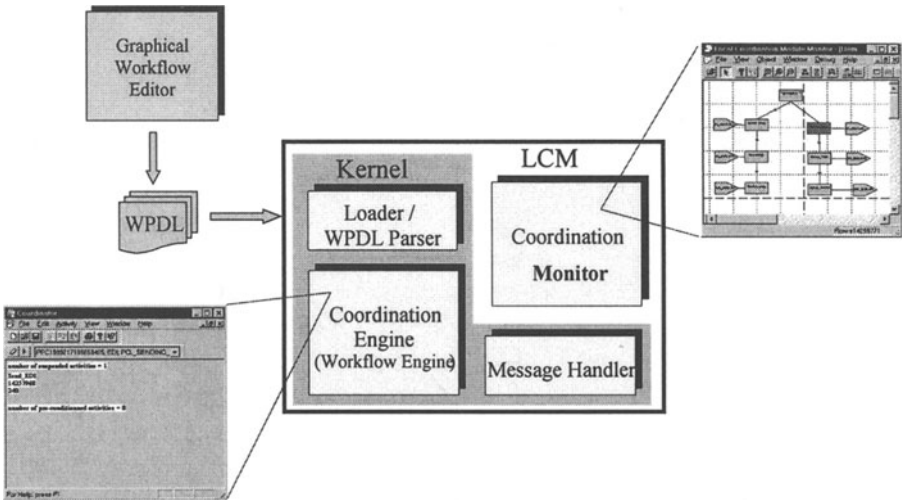

Figure 13 – LCM components

Besides supporting the features described in the PCL coordination mechanisms section, the LCM has the following additional characteristics:

- Supports the execution of several instances of a Core Service simultaneously;
- Supports an interactive (step-by-step) execution of a core service. In this case, the activities are used as breakpoints and the human operator can inspect each single step along the execution;
- Provides a basic level of error detection, including: activities time-out, core services parameters mismatch, auxiliary services parameters mismatch, undefined parameters, communication errors, etc.;
- Offers a visualization of the messages received from/sent to the VE partners;
- Allows the execution trace of one particular Core Service;
- LCM Monitor offers some graphical facilities, such as the zoom in, zoom out and fit;
- Configurable delay execution time;
- List of the last events occurred (flow history);
- Visualization of the activity status based on a color code. The Monitor uses a set of colors to indicate the execution status of an activity, namely: inactive (gray), active (light green), suspended (light blue) and pre-conditioned (magenta). Thus, the human operator can know what exactly is happening inside PCL: what events are being executed and what is the execution status of each one. For example, if one activity has been suspended for a long time, s/he may decide to abort that service.

Figure 13 presents a general overview of the mechanism implemented to support the relation between LCF – the Graphical Workflow Editor, and coordination – the LCM. The Editor receives the workflow models from a human operator and generates a file written in the WPDL format. This file is used by the LCM to build a

finite automata which controls all the activities, transitions and the invoking of the applications defined to support each core service, for that particular company.

HIERARCHICAL COORDINATION IN PRODNET

During the development of PRODNET project, and in order to better support the VE coordination needs, the initial architecture was refined by adopting a three-layered hierarchy. This refinement had some consequences in the PCL coordination model previously presented. This section summarizes the PRODNET hierarchical architecture, including the more relevant impacts in the coordination model.

PRODNET Hierarchical Architecture

In order to better understand and develop the coordination structure within the VE infrastructure, a hierarchical approach seems adequate. On one hand, there are two clearly distinct levels:

(1) The global VE coordination level, and
(2) The coordination of different functions inside each VE-member enterprise in order to fulfil its cooperation events.

Clearly, in level 1 within the VE there is a need for the "coordination" of all its members to accomplish the common VE goal. This coordination can be properly supported through a workflow definition. On the other hand, in level 2, inside a single VE member, there are a large number of inter-module events to be handled to prepare all necessary elements for the "cooperation" of this enterprise with others in the VE, which require the definition of many sub-workflow plans.

Following this approach, the initial architecture of PRODNET can be extended by adopting a 3 layer hierarchical coordination / cooperation structure as represented in Figure 14. In this architecture, in addition to the distinction defined above, there is a separation between the so called "management functionalities" and the basic "cooperation functionalities", as described bellow.

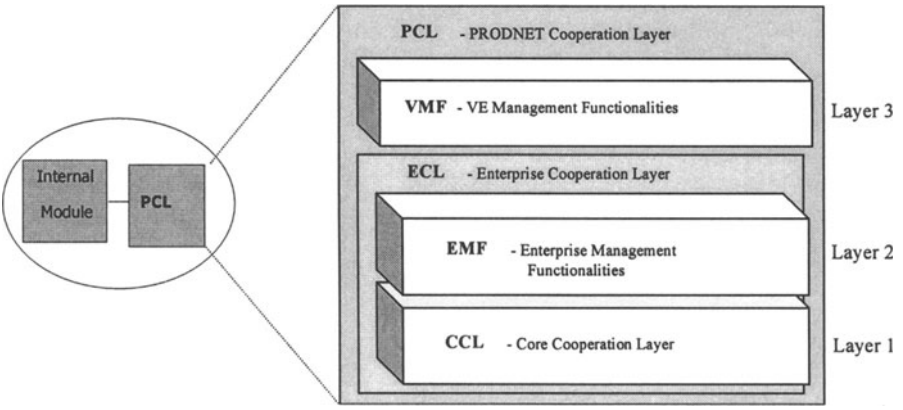

Figure 14 - Hierarchical PRODNET architecture

Each component of this architecture, namely the VMF, EMF and CCL, has the same general structure as represented in Figure 15.

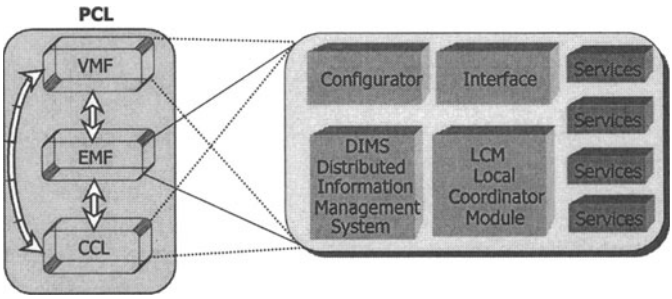

Figure 15 - Structure of the VMF, EMF and CCL components

Core Cooperation Layer (CCL)
The CCL is positioned in the first level (layer 1) and corresponds to the original PCL, as presented in Figure 1. The roles of CCL include:

- Safe communications and admission rights: supported by PCI services.
- Private / shared information visibility (i.e., availability of services and products information to other authorized enterprises): supported by DIMS services.
- Mail box: supported by DIMS (and EDI + STEP, for special messages).
- Deliver mail: supported by PCI (and EDI + STEP, for special messages).
- Federated information access (i.e., no need for centralization of data): supported by DIMS.
- Configurable "behavior" towards a VE environment, supported by both LCF and LCM. The CCL behavior for each kind of event is regulated by workflow models (configured by each company) and executed by LCM, as described in the previous section.

Enterprise Management Functionalities (EMF)
The EMF layer is positioned in the second level (layer 2) of the architecture and is responsible for coordination of the activities at the enterprise level. In other words, the EMF deals with the coordination responsibilities of the enterprise towards the accomplishment of its contracts with the VE and other VE-partners.

In order to cope with the needs of agile decision making to operate in a VE environment, more than a simple re-engineering of legacy systems there is a need for a new and integrated generation of enterprise applications (ERP/PPC, CAD, PDM, and other systems). The exact shape of such future systems still requires further research, since PRODNET was not focused on the development of such concepts. The PPC functionalities should also be integrated with other internal functions of the company (product data management and engineering support functions, for instance). A temporary solution is to build the new layer EMF on top of CCL and establish links with legacy systems.

The EMF layer can also be based on the workflow management architecture, in order to support an easily configurable behavior. The structure of this new layer can be quite similar to the structure of the CCL. In terms of implementation, copies of the same software tools developed in PCL for the LCM and DIMS can be used, but clearly the information models and workflow models are different. The services at this level correspond to extensions of the PPC, CAD, and other legacy systems. In a future scenario, this module could be seen as an "enterprise executor" (or "enterprise operating system") following the CIM-OSA / GERAM architecture.

Virtual Enterprise Management Functionalities (VMF)
The coordination aspects at the VE level are considered in the third level (layer 3). In principle, only the node playing the VE coordinator role will use this layer to monitor, assist, and modify the necessary activities related to the VE goal achievement. The VE Management Functionalities (VMF) will resort to the services provided by the ECL of its node to communicate with the other nodes of the VE. In PRODNET II, a few example services are implemented at this level, such as: *Partners Search and Selection* and *Distributed Business Process Management*. However, this infrastructure is open for the future addition of other services, such as the task announcement edition and configuration, semi-automatic negotiation, electronic contract edition and configuration, as well as other services related to, for instance, collaborative engineering, or digital pre-assembly.

Short term flows / Long term flows
PCL has to cope with some events that can be performed immediately whilst others may take days to be executed. For instance, the process of sending a purchase order can be executed quickly, but the reception of an order acceptance related to that purchase order could take several days. Considering the examples of services supported in PRODNET II, the CCL services tend to have short term execution flows while both EMF and VMF services tend to require long term execution flows.

The extended PCL Services

As already mentioned, the concept of *service* is a key concept in the PRODNET infrastructure. In previous sections only the PCL Core Services were described, however the hierarchical architecture added two new classes of PCL services. The complete list of PCL service classes is the following:

- **PCL Core Services**: related to the Core Cooperation Layer, these services are strongly devoted to support the basic interoperation of an enterprise with a VE environment. Examples of such services are: *Sending a client order*, *Receiving an order acceptance*, etc.;
- **PCL Enterprise Services**: related to the Enterprise Management Functionalities, they are strongly based on the clauses specified in the contractual agreements between companies and include services such as: *Monitoring Cyclical and Periodic Orders Sending*, *Monitoring Order Acceptance*, etc.;
- **PCL VE Services**: related to the VE Management Functionalities, they are strongly related to the VE global coordination including services such as:

Request Production Follow Up Bulletins, Send Production Follow Up Bulletins, etc.

As *PCL Core Services* were previously described, only the other two classes are presented here.

PCL Enterprise Services

At the EMF level, the services are basically related to the ***management of contracts***. This means, each enterprise has rights and obligations to fulfil related to the business relationships with its partners.

An important characteristic found in these services is the control of cyclic and periodic activities. For instance, let us consider the following contract clause: "*company A shall send a Purchase Order to company B, during the first week of each month for the current month as well as shipping forecast for the next month*". This can be viewed as a periodic activity that has to be done each week / month.

Another clause example: "*An individual purchase order becomes valid when company A sends a purchase order and company B accepts it. If company B cannot accept the order, company A must be informed within a period of five working days*". Such clause requires a particular service in order to guarantee that the acceptance of a purchase order is always monitored.

In PRODNET II some simple services to support the *management of contracts* are implemented at the EMF layer. Therefore, some examples of the Enterprise services are the following: Monitor Sending Purchase Order, Monitor Sending Purchase Order forecast, Monitor Receiving Purchase Order, Monitor Receiving Purchase Order Forecast, Monitor Acceptance of Purchase Orders. Other functionalities (services) included at this level are visualization of product models (e.g. INTRAVISION tool) and management of administrative data (e.g. PDM tool) associated to product models. In future, most services offered by the PPC and ERP systems could be made available at this coordination level, but this is out of scope of the PRODNET II project.

Another issue that could be considered at this level is the automatic generation of suggested workflow models, based on a *company profile*. For instance, if a company has a PPC system, it will probably use the order-related services, such as: send/receive purchase order, send/receive order acceptance, send/receive production bulletins, etc. If a company has a PDM editor, it will probably use the STEP-related services, such as send/receive CONDRA messages. In both cases, the workflow editor could automatically suggest an initial set of services.

PCL VE Services

At the VMF level, the considered services are related to the management of the VE itself, as a whole. Examples of such services are the following: Distributed Business Processes (DBP) monitoring, Monitor Sending Production Follow-up bulletins, Monitor Receiving Production Follow-up bulletins, or Request Suppliers Search and Selection.

It shall be noticed that management of contracts may have consequences both at the EMF and VMF levels. The contributions to a VE contract being "processed" by the VE members (distributed business process) are monitored by the VE coordinator

via VMF services. For each VE member, via its EMF layer, only the aspects related to the enterprise itself are handled. For instance, a contract may specify that a client must inform, on a weekly basis, the exact amount of products / parts required for that period according to a general supplying *contract*. A periodic monitoring activity should therefore be supported, both by the client or VE coordinator and by the supplier node, in order to verify / guarantee the fulfillment of this contractual requirement.

As another example, a contract may oblige the supplier to send to its client or VE coordinator, periodic reports on the status of its orders, which also requires a kind of periodic monitoring workflow. *Temporized* and *cyclic* activities are therefore extremely important for these coordination levels.

Interoperation between coordination modules

As shown in Figure 15, there is a coordination module at each level of the PRODNET multi-layered architecture – CCL, EMF and VMF. Therefore, an important mechanism in the architecture is the support for the interaction between these three different coordinators.

At each level, the coordinator plays the role of a 'service supplier' to the level above it, which in turn implies that some actions in the core level can be "connected" to the actions to be performed in both the enterprise level and VE level. For instance, when a client order arrives at the PCL, the *core coordinator* should inform the *enterprise coordinator*, so that the enterprise coordinator can validate some contractual clauses related to that order and / or client.

Figure 16 presents the implementation approach to support the interoperation between coordinators. Each coordinator module provides an API that offers: 1) the definition of the services offered by the module; and 2) a communication process that guarantees the sending of messages to that coordinator. In PRODNET, the APIs are implemented as DLLs (Dynamic Link Libraries) and the adopted communication method is based on sockets or RPCs (remote Procedure Calls). Each API implements a socket client and each coordinator module implements a socket server.

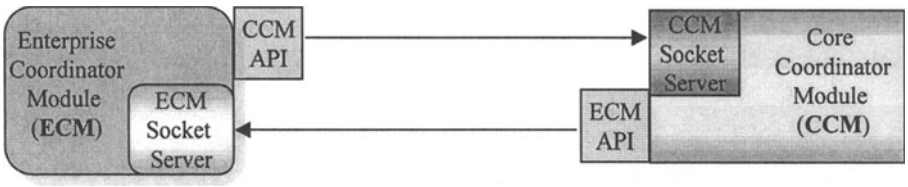

Figure 16 - Interoperation between coordination modules

Configuration issues in the hierarchical architecture

Each level of the hierarchy has some particular configuration needs. The configuration needs at the CCL level were presented in a previous section. At the EMF layer, it is necessary to configure the company environment. Here, a *Company Profile* represents the main characteristics of the company's internal environment

(such as the enterprise applications and other legacy systems to be linked to PCL), about the PCL users inside the company, and the customization of the PCL configurable "features". Related to the configuration of the VE itself there is a *VE Network Directory*, which contains the global VE configuration. This means, all information about each VE an enterprise is involved in, since an enterprise is able to participate in several VEs simultaneously.

At the VMF level, the *VE General Information* represents the information globally related to the VE, such as: VE identification, VE members' identification, VE Common Information (to be accessed by all members), Current VE business processes, and so on. The *VE Coordinator Profile* contains the information necessary to support the VE coordinator to accomplish its global coordination task. The *VE members profile* covers the information related to the VE partners. The *VE relations* is mainly concerned with the visibility level of each VE member and the VE coordinator, in terms of access rights, privileges, etc. Some examples of required information are: access rights – privileges between VE Coordinator and VE member, tasks to be performed by a member (for instance, a VE member has to periodically send the quality information to the VE Coordinator), some global clauses on the contracts signed between the VE coordinator and each VE member, etc.

Finally, a relationship involving VE members is considered since it provides some additional information that can improve the knowledge about the VE environment. Some examples of relevant configuration related to the VE members are: access rights (privileges between two VE members), members interoperation – configuration of cooperative tasks, etc.

CONCLUSIONS AND FUTURE WORK

Based on the workflow reference model from the WFMC, the PCL coordination mechanisms described in this chapter presents the required features to cope with the coordination issues in the PRODNET hierarchical coordination architecture. The needs identified at each level are clearly supported in the model. Both modules involved in the model implementation were also described including some implementation aspects.

Several other projects including NIIIP-SMART (Barry et al., 1997, 1998), WIDE (Chan et al., 1998), and VEGA (Bakkeren et al., 1997), also adopted a workflow-based coordination approach and some of them developed specific workflow engines and workflow editors. There are, however, several unique features proposed by PRODNET II such as the hierarchical coordination architecture and the support for loops and cycles inside workflow models as a basis for contract management.

Considering that PRODNET II is a three-year long project and the scope envisaged for PCL is very ambitious, there is still some work to be done in order to fully implement the coordination model presented here. Nevertheless the essential features that validate the model are already implemented and validated in the PRODNET demonstration scenario. Some examples of future work are:

- Implementation of a stronger mechanism to support the interoperation between coordinators in the hierarchy.

- Support to dynamic changes in the workflow models.
- Further research on workflow-based analysis techniques.
- Extensive analysis and development of the required functionalities at the EMF and VMF levels.
- Implementation of configurable mechanism to support workflow-based error handling.
- Provide a library of workflow model templates to make the configuration task easier for the human operator.
- Reinforce the integration between PCL coordination mechanisms and Enterprise Applications, through an open PCL API .

Acknowledgements

This work was funded in part by the European Commission, Esprit program, and the Brazilian research council (CNPq).

The authors also thank the valuable contributions from the consortium partners: CSIN (P), ESTEC (P), HERTEN (BR), Lichen Informatique (F), MIRALAGO (P), ProSTEP (D), Uninova (P), University of Amsterdam (NL), Universidade Federal de Santa Catarina (BR), and Universidade Nova de Lisboa (P).

REFERENCES

1. Camarinha-Matos, L.M.; Lima, C.; Osório, L. - *The PRODNET platform for production planning and management in virtual enterprises.* Proc. of ICE'97 4th Int. Conf. on Concurrent Engineering, Nottingham, October 8-10, 1997.
2. Camarinha-Matos, L.M.; Pantoja Lima, C. - A Framework for Cooperation in Virtual Enterprises, Proceedings of DIISM'98 - Design of Information Infrastructures Systems for Manufacturing 1998, Fort Worth, USA, May 98.
3. WfMC - Workflow Management Coalition (1994) - The Workflow Reference Model - Document Nr. TC00 - 1003, Issue 1.1, Brussels Nov 29, 1994
4. WfMC - Workflow Management Coalition (1998), Interface 1: Process Definition Interchange Process Model, Document Number WfMC TC-1016-P, Issued on August 5, 1998.
5. OMG – OMG document number. Workflow Management Facility, OMG Document Number WfMC bom/98-06-07, Issued on July 4, 1998.
6. Bakkeren, W.; Zarli, A.; Debras, P.; Schulz, K.; Köthe, M.; Korsveien, S. - A Model of Worklfow – Specification of a Model for the Definition of Workflows in Virtual LSE Enterprises; Public Report from Esprit Project 20408 -VEGA , issued by Jan 97.
7. Chan, D.; Vonk, J.; Sánchez, G.; Grefen, P.; Apers, P. - A Specification Language for the WIDE workflow Model.
8. Barry, J.; Aparicio,M.; Gilman, C.; Ramnath, R. at al - Integration of design and manufacturing in a virtual enterprise using enterprise rules, intelligent agents, STEP and workflow, Architectures, Networks and Intelligent Systems for Manufacturing Integration, B. Gopalakrishnan, San Murugesan, Odo Struges, Gerfried Zeichen, Editors, Proc. of SPIE Vol 3203, pp.160-171 (1997).
9. Barry, J.; Aparicio,M.; Gilman, C.; Ramnath, R. at al - NIIIP-SMART: an Investigation of Distributed Object Approaches to Support MES Development and Deployment in a Virtual Enterprise, 2nd Int. Enterprise Distributed Computing Workshop (EDOC'98), 2-5 Nov 1998.
10. OMG – OMG document number. Workflow Management Facility, OMG Document Number WfMC bom/98-06-07, Issued on July 4, 1998.
11. van der Aalst, W. - The Application of Petri Nets to Workflow Management.
12. Alonso, G. at al, - *The WISE approach to Electronic Commerce,* http://www.inf.ethz.ch/department/IS/iks/research/wise.html, Feb 15, 1999.