# Methods to Support Co-operative Product Development

*R. Anderl[1], J. Bumiller[2], M. Krastel[1], K. Schiemenz[1], M. Stupperich[2]*

*[1] Darmstadt University of Technology*
*Department for Computer Integrated Design (DiK)*
*D-64827 Darmstadt, Petersenstr. 30, Germany*
*{anderl, krastel, schiemen}@dik.tu-darmstadt.de*

*[2] Daimer-Benz AG, Research and Technology,*
*D-89013 Ulm, Postfach 2360, Germany*
*{bumiller, stupperich}@dbag.ulm.DaimlerBenz.com*

**Abstract**

World-wide distributed product development processes and manufacturer supplier co-operation has become more important in recent years. This requires support for collaboration on the one hand and the administration of the generated data on the other hand. In this paper methods for determination of co-operation needs during distributed product development processes are presented. The particular co-operation situation which shall be supported can be classified by given criteria. New data management methods to support a distributed co-operation process are shown. A mapping scheme is provided that allows to assign these new data management methods to the classified co-operation needs.

**Keywords**

**Collaboration, Co-operation, CSCW, EDM, Engineering Data Management, PDM, Product Data Management**

G. Jacucci et al. (eds.), *Globalization of Manufacturing in the Digital Communications Era of the 21st Century*

# 1 INTRODUCTION

Today's world-wide competition and the need for faster product cycles require highly efficient distributed product development processes. This includes the processes in the producing company as well as in manufacturer-supplier chains especially in development and simultaneous engineering co-operations. Individual tasks and processes are effectively supported by various CAx systems. The effective management of the generated data or even more the management of all information during the product development process becomes a major challenge. State of the art product data management (PDM) systems partly fulfil the requirements. The challenge becomes even greater for co-operative product development processes distributed over different locations. Furthermore distributed processes require a special support for the collaboration between different locations.

In our approach called MUVER (multimedia support for distributed engineering teams) we are working on a systematic support of distributed product development processes. This includes methods for synchronous as well as for asynchronous co-operation. In this contribution we are presenting a systematic approach to use data management and co-operation methods and tools to support distributed engineering teams. Therefore we first discuss the requirements of distributed product development processes with respect to data management and co-operation needs. Then we present the classification of co-operation developed in this project. In the next part we discuss the functionalities of state of the art PDM-systems and propose methods derived from other data management systems. The linking of the characteristics of co-operations and data management methods leads to a mapping scheme to support individual co-operations by PDM methods. The paper concludes with a short summary and an outlook on further work.

# 2 REQUIREMENTS OF DISTRIBUTED PRODUCT DEVELOPMENT PROCESSES

Distributed product development processes faces two major challenges:

(1) the management of the generated product data created at different locations and
(2) the support of the resulting co-operation needs.

State of the art product development processes are supported by various CAx systems, like CAD, FEM and simulation systems. The systems are supporting the development process very efficiently in their own application context and help to improve time to market as well as product quality.

With the use of these application systems various types of product data are generated. These data are not only used isolated in their authoring system but also are reused along a part or even the whole product development process. In this progress the data are possibly converted into other formats with different versions. This results in a lot of dependencies between the data. The efficient management of the various data types and the dependencies between them is a requirement for an effective process. In this context management of the data means functionalities like data storing, data archiving, efficient search and retrieval functions, access control

mechanisms etc.. Requirements for the management of application systems data are quite well documented (Abramovici e.a. 1997, Anderl e.a. 1996, Encarnação e.a. 1990).

In distributed product development processes the requirement for product data management is becoming even stronger. The goal of a distributed data management is the active support of co-operation of several persons on a common database. Additionally to local development processes it is necessary to be able to access the application systems data transparently through the current location. Further on rigid access control methods impede the dynamic work processes in teams and result in less productive work.

The second major source of requirements is the need of co-operation support. On the one hand methods and tools for the support of co-operation are required. On the other hand the data generated by these supporting tools shall be managed in an appropriate way.

One source for the determination for co-operation needs in distributed product development processes is the description of the interdependencies of product components in a matrix (Beitz e.a 1996, Seeliger e.a. 1997).

All components of the selected product structure level are laid on both axis of the matrix while the component dependencies are given in the matrix. An example is shown in Figure 1.

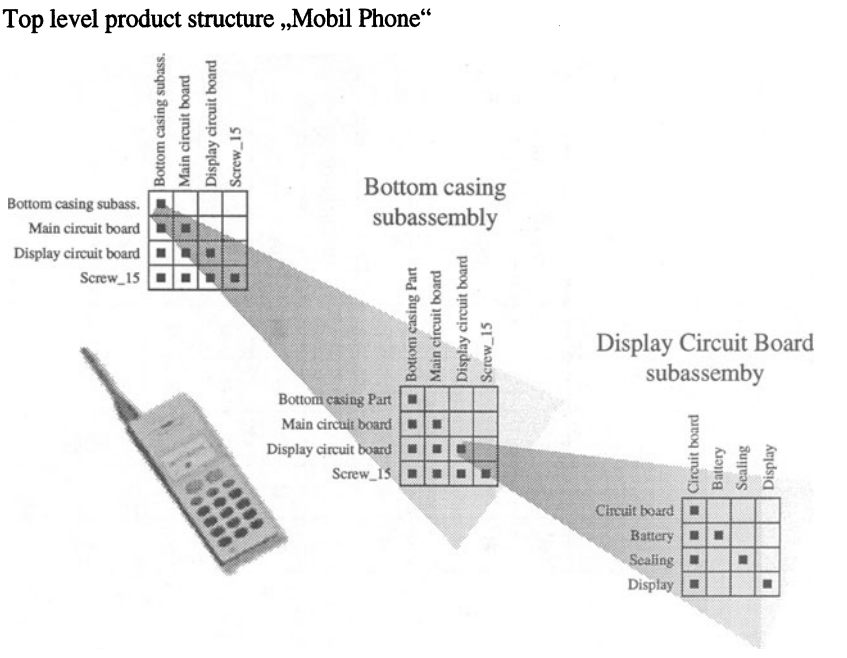


Figure 1: Matrix of dependencies between product components

It is also possible to enter different types of dependencies, like for example geometrical, functional or connecting dependencies. The method contains the possibility to use different hierarchical levels for the explosion of the entire product. It is possible to represent dependencies between complete subassemblies on a high, as well as between individual parts on a detailed product structure level. Beside the allocation of different dependency types it is furthermore possible to illustrate the production place of the respective component.

Product development processes and in particular the design process cannot be mapped as rigid workflows with a fix sequence of activities. A workflow can be defined only on an abstract level, e.g., the temporal project plan with milestones. Therefore we chose a different way and look on interactions in distributed processes. This will be our second source to determine co-operation requirements. During the entire product development process a lot of interactions occur, which shall be supported in an appropriate way. Organisational aspects including the division of the main task into subtasks, the co-ordination of the distributed working processes and the joining of single results to one overall result require interaction processes between the involved persons. In particular this is valid for the design process, because the rate of creative activities is very high. Furthermore the iterative sequence of the design work causes changes and discussions on the effects of the changes. The use of creativity techniques to identify solutions in distributed environments should also be supported.

Table 1: Interactions

| | *Cause* | | *Trigger* | | *Control* | | *Relation* | | |
|---|---|---|---|---|---|---|---|---|---|
| *Interaction* | *coordination* | *informativ* | *reaktiv* | *aktiv* | *arranging* | *autonomous* | *1:1* | *1:N* | *N:M* |
| *Inquiry* | | ■ | | ■ | | ■ | ■ | ■ | |
| *Announcement* | | ■ | | ■ | | ■ | ■ | ■ | |
| *Order* | | ■ | | ■ | | ■ | ■ | ■ | |
| *Statement* | | ■ | | ■ | | ■ | ■ | ■ | |
| *Suggestion* | ■ | ■ | | ■ | | ■ | ■ | ■ | |

...

In addition, spontaneous incidents and spontaneously occurring questions require functions to support informal and 'ad-hoc' communication. For the identification of the variety of possible interaction we extended a list provided by Dürr (1994). We

classified the interactions by defining several general types. A part of the list is shown in Table 1.

With the knowledge of process interactions it is now possible to classify and in a further step to support co-operation in distributed product development processes best.

## 3 CLASSIFICATION OF COOPERATION

To support distributed product development processes with PDM and CSCW methods a classification of co-operation needs in processes was made. The criteria of this classification are based on the interactions presented in Table 1.

In a first approach co-operation could be divided into asynchronous and synchronous sequence. The type of sequence is often influenced by the 'time factor' in co-operation. In a time critical co-operation a synchronous sequence helps to reach the aim faster. The 'time factor' is also important in an information context because the required information affects the kind of data organisation (see chapter 4.2).

To provide co-operation support the number of participants and the cardinality of communication channels is important, because common CSCW tools only support a particular cardinality (e.g., audio 1:1, mail 1:N, file transfer 1:1 etc.). In our classification we differentiate between partner co-operation (1:1, bilateral), co-operation in a small group initiated by one person (1:N) and co-operation in a large group with an interaction network (N:M). The determination of co-operation partners before starting and the constant participation during the co-operation are further characteristics. These factors restrict the usable CSCW tool, because the needed hard- and software has to be available for every partner involved in the process.

Since engineering processes are not performed deterministic, two different ways to initiate a co-operation exist. On the one hand there is a planned process and therefore a planned co-operation. On the other hand ad-hoc co-operation occurs, e.g., caused by errors or questions in the process. Similar to the initiation, the needed information within the co-operation is different. In a process initiated co-operation the information needs are known but in an ad-hoc co-operation they are often undefined. Other characteristics are the various structuring levels of co-operations. The basis of a full-structured co-operation is a defined workflow, whereas unstructured co-operation appears in group discussions. In dependence of these different types of co-operation several methods of access right assignment and concurrency have to be supported (see chapter 4.2).

In order to determine the supporting effort, knowledge about the repeating rate of the co-operation is of interest. To find the appropriate method the purpose of co-operation is also important. The effort required to support a co-operation with co-ordination purposes are higher than for a co-operation with informative character. The problem solution character of a co-operation can reach various levels of complexity. From co-operation in routine tasks with low complexity, over medium complex up to group co-operation tasks performing creative work. Finally the effort required to support co-operation grows up with its complexity.

Further on the communication medium is essential for the selection of CSCW tools. Possible mediums are a physical file in a file transfer, plain text in an e-mail, documents in shared editors, graphics on a whiteboard, speech in an audio transmission, live-video in a video transmission and application sharing in a shared workspace.

Last but not least the co-operation can be classified on the technical level by the available bandwidth of the network. Table 2 shows a summary of the classification.

Table 2: Criteria for the classification of co-operation needs

| *Co-operation criteria* | *values* |
|---|---|
| Sequence | synchronous, asynchronous |
| Time-critical | yes, no |
| Information actuality | high (real time), small |
| Cardinality | 1:1 (partner), 1:N (middle number), N:M (large number) |
| Partner | not determined, determined, unchanging, changing |
| Initiation | ad-hoc, deterministically |
| Structure | full-structured, unstructured |
| Repeating rate | low, medium, high |
| Purpose | informative, coordinative |
| Problem solution character | creative, analytic, routine tasks |
| Information needs | unknown, determined |
| Medium | files, plain text, documents, graphics, speech, video, application sharing |
| Available network bandwidth | large (e.g., LAN, AT, several ISDN B-channels), medium (e.g., 2 ISDN B-channels) small (e.g., modem connection, 1 ISDN B-channel) |

# 4 PRODUCT DATA MANAGEMENT IN DISTRIBUTED PRODUCT DEVELOPMENT PROCESSES

In this chapter we describe on basis of the requirements possible solutions for product data management for distributed product development processes. In the paper we use Product Data Management (PDM) and Engineering Data Management (EDM) synonymously for systems managing all data generated in the product development process. We shortly describe common methods of Product Data Management systems and the deficits with respect to distributed product development and show some new PDM methods derived from other data management systems.

## 4.1 Common PDM Functions

Product Data Management Systems are mainly used in technically oriented product development processes. Current PDM systems contain the following functions based on a classification by Eigner (1991):

- administration of items,
- administration of process sequences,
- administration of privileges and
- administration of files.

The administration of items covers three classes of items: projects, articles and documents. In this approach articles represent both the products themselves as well as the assemblies and parts. The administration of these items is done through meta data. Thereby a multiplicity of functions for classification and various search possibilities are available.

In the context of item administration it is also possible to handle and visualise the structural relations between the items. Those can be done for both items of a class and class-spreading. An example of the administration of relations within a class of items is the administration of the product structure. Class-spreading, e.g., a product component can belong on the one hand to different projects and can be on the other hand described by several documents (e.g. 3D-Modell, drawing, calculation model, etc.). The administration of different item versions is also covered in this context.

In the context of process sequence administration the product development process flow and the representation of the product development history can take place. However only rigidly given process cycles can be illustrated and administered in the form of release flows. Pre-defined engineering change requests can be managed in the same way.

The administration of privileges allows to assign access rights on administered data to users and user groups (e.g., design manager, design team, individual designer). Also status dependent modifications of access rights in context of pre-defined process cycles can be visualised.

The administration of files means storing the product related models and files in their native file format in so-called electronic vaults. The access to the files is provided by the PDM system only and can be managed by functions for administration of privileges. Currently available PDM systems are hardly supporting a geographic distribution of meta databases and electronic vaults. Further concepts, like the integration of HTML interfaces, external e-mail systems and intelligent agents for the replication of data which are already available in GroupWare systems are presently under development. However the area of PDM especially with respect to distributed product development processes is still a major area of research activities (Abramovici e.a., 1998). Especially the combination of product data management with group- and co-operation support, as known from GroupWare-Systems, is still neglected.

## 4.2 New PDM methods

In order to support co-operation, new functionalities have to be implemented into product data management systems. In the following, concepts derived from the three areas GroupWare, Data Warehouse and distributed data bases are presented.

*Concepts of GroupWare systems*

GroupWare systems are using concepts to handle distributed data needed for co-operations (Table 4). Consistency questions become particularly critical whenever synchronous or asynchronous co-operation on the same data takes place. In the CSCW area special attention is paid to the flexibility of the users. Therefore different concepts were developed to handle concurrency of user access. Concurrency control in general offers the possibility that several users can operate on the same data. An optimistic concurrency control assumes that two users never access and modify an object in the database at the same time. The consistency of data is thus not always guaranteed, but it is possible to achieve the highest degree of parallelism because no restrictions regarding the access are made. With the pessimistic procedures the consistency of data is guaranteed every time (e.g., simple lock mechanisms). This means that it is only possible to get read access to an object in work by another user, hence there is no possibility to work parallel with the same data.

The concepts of replication are a third focus in the area of GroupWare systems. Depending on actuality and volume of data in the network the period of replication and the replication method can be chosen.

Table 3: Data management concepts of GroupWare systems

| *Concepts* | *Possible solution* |
|---|---|
| Access right assignment | statically, dynamically |
| Concurrency control | simple lock, optimistic procedures, floor passing technique, locks with notification, procedures with voting, version based protocols |
| Replication | none, complete, partly, only modification |

*Concepts of Data Warehouse systems*

The integration of most different data sources and the availability of information for the user within a uniform user interface are the main targets of a enterprise wide Data Warehouse system. To integrate the locally produced data in a Data Warehouse different migration concept were developed. With the migration by refurbishing (Pull) the data in the Data Warehouse are overwritten periodically, therefore no information about data history is stored.

Table 4: Data management concepts of Data Warehouse systems

| *Concepts* | *Possible solution* |
|---|---|
| Granularity of data | constant, gradual |
| Data archive | central, peripheral, none |
| Migration | refurbish, update, publish |

The migration by updating completes the data following the modifications which took place since the last update. The technically most difficult method is the migration by publishing (Push). Modifications in the source data are mapped automatically in the Data Warehouse. Also concepts to store different granularity of data and to archive older data are used in Data Warehouse systems.

*Concepts of distributed data bases*

The concepts of distributed data bases are information orientated. They are concerned with the system organisation, the consistency, the integrity, the caching, and the concurrency control of data. There are a lot of procedures to regulate concurrency access. In the following table only those procedures are mentioned which could also support co-operative work.

Table 5: Data management concepts of Data Warehouse systems

| *Concepts* | *Possible solution* |
|---|---|
| System organisation | physically central, data base, client/server, distributed |
| Integrity | semantic, relational, operational |
| Recovery | back up, snap shot, journal file |
| Cache | central cache, peripheral cache, no cache |
| Concurrency control | ESCROW protocol, lock mode by "Unland", token based procedure, nested transactions |

## 5 MAPPING SCHEME OF COLLABORATION NEEDS AND DATA MANAGEMENT METHODS

With the classification of co-operation provided in chapter 3 and the described new PDM methods in chapter 4 it is possible to create a mapping scheme with types of co-operation on the one and PDM methods one the other axis. Table 7 shows a part of the full scheme. The entered symbols '■' mean that a co-operation which fulfils a co-operation criteria (row) is supported by a particular PDM method (column).

With the aid of the developed mapping scheme it is now possible to select specified PDM methods for a particular collaboration situation in a co-operative product development process.

The methodical application of this scheme is shown in Figure 2. First the interesting process chain has to be specified. By selecting the particular process which should be supported with PDM methods the information and co-operation requirements have to be identified. Methods to determine co-operation needs were shown in chapter 2. In a second step the needs can be classified with the criteria shown in chapter 3. The scheme shows which PDM method supports a particular co-operation in an appropriate way. As a conclusion this is also the best PDM method to support the regarded process. If the used PDM system has no function to fulfil the method it is a new requirement to the PDM system in order to optimise the process chain.

Table 6: Part of the mapping scheme

| Co-operation criteria | | PDM methods: Concurrency Control: optimistic | Concurrency Control: version ba. | Concurrency Control: pessimistic | data retention: central | data retention: distributed | assign of access rights: static | assign of access rights: dynamic | ... |
|---|---|---|---|---|---|---|---|---|---|
| Flow | Synchronous | | | | ■ | | | ■ | |
| | Asynchronous | | ■ | | | ■ | ■ | | |
| Structure | Full-structured | | ■ | ■ | | ■ | ■ | | |
| | Unstructured | ■ | ■ | | ■ | | | ■ | |
| Information needs | Unknown | | | | ■ | | | ■ | |
| | Determined | | ■ | | | ■ | ■ | | |
| ... | ... | | | | | | | | |

## 6 PROTOTYPE IMPLEMENTATION

In the following a software prototype based on one method identified by the mapping scheme is shortly described. The method was implemented in the EDM system Metaphase® from SDRC (Structural Dynamics Research Corporation).

We chose for this example the 'version based concurrency control' method (Dürr, 1994). This method is suited to support a co-operation which is characterised in the following way:
asynchronous sequence, not time critical, N:M cardinality, structured or unstructured, low, medium or high repeating rate and small available network bandwidth.

The version based concurrency control is based on the exchange of preliminary product statuses. The statuses were described as versions with an array of particular attributes, which is called object specification (e.g., mass, dimensions (length, height, depth), connection dimension ...). If a person needs information about the product he or she specifies an array of needed attributes, which is called user specification (e.g., a marketing person is only interested in the mass of the product and not in the connection dimensions). The system checks if the current version fulfils the required attributes at the moment of request with the following expression:

*user specification ⊄ version specification.*

If the expression is true and the attributes are already specified, the implemented functions allow to check out the product in a preliminary status. If it is false an error message will be sent to the requesting person.

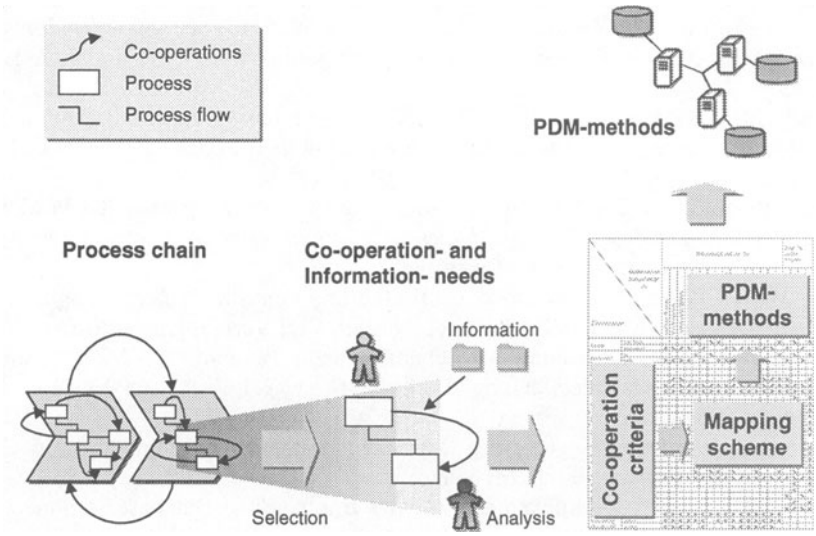


Figure 2: Application of the mapping scheme

## 7 CONCLUSION AND OUTLOOK

In this paper we presented how a distributed product development process can be supported with new methods of product data management systems. The developed scheme can be used to select particular data management methods to support existing product development processes. Additionally the scheme offers the possibility to identify actually not implemented data management methods to support distributed processes. The identified methods can be implemented in form of new applications or new components of available PDM systems.

A further aim is the specification of an optimised co-operation infrastructure and the realisation of a general framework. This framework is an architecture to realise a transparent and system independent communication of objects with different services (Anderl e.a., 1998). One part of this architecture is a PDM system with support for co-operation as described in this paper.

## 8 REFERENCES

Abramovici, M. e.a. (1997) Application of PDM technology for Product Life Cycle Management in *Life Cycle Networks* (eds. F.-L. Krause & G. Seliger) Proceedings of the 4th CIRP International Seminar on Life Cycle Engineering, Chapman & Hall, Berlin.

Abramovici, M. e.a. (1998): Supporting distributed Product development Processes with PDM in: F.L. Krause: *New Tools and Workflows for Product development (ed.* F.-L. Krause) Proceedings of the CIRP Seminar 1998, Chapman & Hall, Berlin.

Anderl, R., Katzenmaier, J. (1996) Trends in der Produktverwaltung in *Engineering Datenmanagment* Industrie Managment Special 1996/97, GITTO Verlag.

Anderl, R., Bumiller, J. e.a. (1998) Multimediale Unterstützung verteilter Produktentwicklung in *Tele-CAD: Produktentwicklung in Netzwerken* (eds. R. Anderl e.a.) Tagungsband CAD '98, Darmstadt.

Beitz, W. e.a. (1996) Eine Methode zur Einführung von Simultaneous Engineering in der Konstruktion in VDI Bericht Nr. 1289, VDI Verlag, Düsseldorf.

Dürr, Martin (1994) Koordinationsmechanismen für Teamarbeit – Modellbildung und Datenbank- Unterstützung. Fortschr.-Ber. VDI Reihe 10 Nr.296, VDI Verlag, Düsseldorf.

Eigner, M. e.a. (1991) Engineering Database: Strategische Komponente in CIM Konzepten. Hanser Verlag, München, Wien.

Encarnação, J. L. P.C. Lockemann (1990) Engineering Database: Connecting Islands of Automation through Databases. Springer Verlag, Heidelberg.

Seliger, G. e.a. (1997) Zusammenarbeit bei der Entwicklung komplexer Produkte, Konstruktion 49, 37-41.

## 9 BIOGRAPHY

Prof. Dr.-Ing. R. Anderl has been head of the computer integrated design (DiK) at the faculty of mechanical engineering, Darmstadt University of Technology since 1993. Dipl.-Ing. M. Krastel and Dipl.-Ing. K. Schiemenz are working as research assistants at the department DiK in the area of CAD and PDM for engineering teams in distributed product development processes.

Dipl.-Inf. Johannes Bumiller studied computer science in Karlsruhe. He joined Daimler-Benz research 1988. He spent one year at MIT Center for Co-ordination Science working on Computer Supported Co-operative Work, especially process support. Michael Stupperich studied computer science in Karlsruhe. He joined Daimler-Benz research 1993 and is working on Computer Supported Co-operative Work, especially process support and information spaces.