A peer to peer based approach to collaborative product engineering

Patrick Stiefel, Jörg P. Müller
Institute of Computer Science, Business Information Systems Research Group, Julius-Albert-Str. 4, 38678 Clausthal-Zellerfeld, Germany
patrickstiefel@gmx.de
mueller@in.tu-clausthal.de
http://winf.in.tu-clausthal.de

Abstract. In this paper we present key elements of a peer to peer (P2P) based software architecture for integrated and collaborative product engineering. We argue that distributed and cross-enterprise product lifecycle management can benefit from the availability of decentrally managed product model repositories. We present the overall system architecture of a platform called Peer-to-Peer Product Collaboration Platform (PCP). Our focus in this paper is on the design of a decentral product model metadata repository and registry based on a peer-to-peer resource management platform, and on a prototype implementation of an Eclipse-based tool for editing and browsing the metadata repository.

Key words: peer to peer (P2P) architecture, collaborative product engineering, knowledge repository, product data management, product lifecycle management

1 Introduction

In the late 1990s, the term Product Lifecycle Management (PLM) was coined to describe a business activity to manage a company’s products all the way across their lifecycles in the most effective way. [...] PLM helps getting products to market faster, and enables better support of customers’ use of products. [13]

PLM provides a shared platform for creation, organization and dissemination of product related knowledge and includes product-oriented processes covering concept, design, fabrication, assembly, testing, delivery, and disposal of products. PLM extends product data management (PDM) out of engineering and manufacturing into other areas like marketing, finance and after sale service and at the same time, addresses all the stakeholders of product throughout its lifecycle.
Over the past few years, numerous PLM software vendors have established presence in the market, and the market itself has been growing considerably\textsuperscript{1}. Before the background of globalization, global outsourcing, and market pressures for decreasing product time-to-market, PLM faces new challenges. The conception, design, and manufacturing of products is becoming a largely distributed, cross-enterprise endeavor; large distributed product developer teams need to work together in devising new and advancing existing products. Companies that compete in certain market areas will co-operate in others to jointly create and market products. This creates the challenge of extending PLM functions across companies to allow all parties involved in PLM processes to create, publish, access, and update PLM-related knowledge, and to be able to do so in the presence of heterogeneous PLM solutions used in difference companies, of geographic distribution, of changing cooperation partners (especially at the lower tiers of the supply network, where Small and Medium Enterprises are concerned), and of confidentiality and data protection requirements \cite{8,10}.

The working assumption underlying our research is that for supporting collaborative and cross-enterprise PLM processes, it is beneficial to employ methods and architectures enabling decentral management of product-related resources. These methods will aim at enabling loosely coupled interaction between changing partners in a decentral environment, where traditional hierarchical client-server based architecture may not be applicable because storing e.g. all product models and model components in a central server, or orchestrating and monitoring all product-related processes from one central point may be infeasible or – for reasons of competition – undesirable. We further claim that peer-to-peer (P2P) computing offers concepts, protocols, and tools that can are suitable to proceed to the vision of collaborative, cross-enterprise PLM. The goal of our research is to come to an understanding of the potential, the trade-offs and the limitations of P2P technologies in the PLM context, and to provide new methods, protocols, and a software architecture suitable to enhancing existing PLM solutions with the ability for decentral management.

In this paper, we report ongoing research aiming at a platform for decentral management of PLM-related artifacts: The Peer-to-Peer Product Collaboration Platform (PCP). Focusing on an application scenario from collaborative product design, we describe the underlying software architecture of PCP (see Section 3) and give a more detailed account of a core component of our work, the Decentral Product Model Metadata Repository (see Section 5). We present the prototypical implementation of a tool based on the Eclipse framework (see Section 6, which allows us to share, find, and edit metadata records for dedicated product models, and ultimately share, access and process the product models (e.g. 3D-CAD models) themselves, by integrating with different CAD / PDM systems and model repositories.

\textsuperscript{1} For the first time, in 2005 PLM revenue in the German speaking region were reported to have exceeded two billion Euro.
2 Background

2.1 Requirements of cross-enterprise product-related collaboration

The idea of looking at PLM and PDM processes as collaborative processes is not new [4, 8]. Today’s leading PLM vendors (e.g., Agile Product Collaboration) advertize their solutions with a strong focus on the term "collaboration".

Fig. 1. Conceptual architecture for collaborative PLM according to [10]

In [10], Scheer et al. propose a conceptual architecture for collaborative PLM/PDM, which is illustrated in Figure 1 using the example of an OEM-supplier relationship. The vision of collaborative PLM as illustrated in Figure 1 requires providing PLM processes and functions e.g. using techniques from Computer Supported Cooperative Work (CSCW) with methods for Co-Viewing or Co-Modelling of technical drawings (2D/3D), including the need for communication and data exchange for effective teamworking. Decisions and defined work packages have to be documented; in addition, security mechanism are needed to provide confidentiality and access control. Most importantly, collaborative PLM sets up a huge integration and interoperability challenge, i.e. integrating different formats, models, and processes supported by different PDM systems by means of a Collaboration Platform (see Figure 1).

A core function of the platform is to provide a unified product model repository with the basic functions enabling creation, storage, sharing, and retrieval of product information across distributed teams and heterogeneous PDM systems. There are different approaches possible to the actual organisation of the Collaboration Platform. The most obvious (and most frequently used) architectural approach is to implement the collaboration platform by means of a central (possibly distributed) system including the required databases and PLM services.

\footnote{http://www.agile.com/solutions/pc/}
Such an architecture has a number of advantages, including fast, barrier-free communication, efficient access and operation, and good manageability (e.g. in terms of a security architecture). However, in the remainder of this section we argue that in many cases this architecture is either not the best option, or not even applicable at all.

2.2 Need for decentral collaboration support

Product models are constructed on a day-to-day basis by using CAD software and handled in PDMS in companies’ product development departments worldwide. In a global market with rapidly changing partnerships and Virtual Enterprise ventures, there are disadvantages associate with maintaining these product models or individual components in any type of central system. First, it may be technically infeasible due to the huge set of product models to be maintained; second, the fact that these product models belong to (and are key assets to) individual companies makes the task of determining which partner is to host such a central system hard, and (depending on the market settings) politically infeasible to solve. Third, the everchanging structure of the global PLM networks make administration and management of central platforms a very difficult task.

In particular at an early phase of product conception and design, we believe that a more decentral architecture which enables loosely coupled interaction among peers and which provides tools and methods enabling companies to discover PLM-related artifacts (such as partners, skills, requests for quotation, product models) has huge benefits over a centrally organized one. As cross-company development teams are being set up to devise a product and as product models become more concrete and elaborated during the process, this decentral architecture may migrate into a more centrally managed group collaboration architecture allowing more efficient communication and data access / sharing.

The work presented in this paper aims at enabling product development partners to design a new product or redevelop an existing one in a decentrally organised collaborative development environment. What we are looking for is a product information integration system to be used by (at least) the two groups of actors listed in Figure 1: product developers and suppliers. Users usually prefer working using the PDM and CAD applications they are familiar with [3, 5]. Also, from a point of security there are advantages in not having to offer sensitive business knowledge in a central organised product development forum [4].

2.3 Peer-to-peer platforms for product collaboration

Peer-to-peer (P2P) technology [11] provides a number of features which make it attractive to implement communication, data handling, and collaboration in a decentral fashion. Centralized PDMS that serve many clients bear high system cost. A service-oriented P2P architecture can help reducing and sharing the cost (hardware and software licenses) across the participants. Furthermore, all peer nodes are directly connected: There is no central instance for handling communication in the development process. This results in less latency – unfiltered
messaging is enabled in real time. Supporting ad-hoc environments, where members join and leave the network dynamically, is a key feature of collaborative work. P2P satisfies this criterion considering changes in the group of participants [2]; it is appropriate in cases where users do not wish to rely on (or cannot agree on) any centralised or third-party service provider. P2P systems support this level of autonomy because they require local nodes to decide about resources published to other users in due time [9].

3 PCP: Peer-to-Peer Product Collaboration Platform

In this section, we give an overview of the PCP platform. It enables forming a virtual community of product developers and part suppliers for collaborative engineering based on a decentral P2P Resource Management middleware.

3.1 Application scenario: Collaborative product development

The following collaborative product development scenario is used to illustrate the PCP platform: Some of the needed components (parts or subassemblies) may already exist at PDM systems of the manufacturing company, while others do not. A (re-)development process relies on suppliers’ domain knowledge. Our use case allows the OEMs product developer to query suppliers’ product model databases for models by issuing requests for proposals to the virtual community.

![Fig. 2. UML use case diagram for the application scenario.](image)

In this application scenario we regard two actors: OEM product developers and suppliers. Assume a developer intends to design a Lego house, consisting of different components (ultimately coming down to models of individual building bricks). The developer needs to identify appropriate components and
query its suppliers. Thus he publishes requests for components in the P2P network. In the UML use case diagram in Figure 2 this action is generally shown in the use cases numbered 1-3. After having been informed about new existing requests in the network (use case 4), the suppliers may generate corresponding proposals by using their local model repositories (5) and publish them (6,7). The product developer is then able to view existing proposals (8,9) and to access the required 3D models. It is important to note that our approach is to manage product model metadata in the P2P network, whereas the product models themselves may be stored in any system or database outside the P2P network.

3.2 Overview of the architecture

Figure 3 shows the top-level architecture of PCP. The core of the platform is the PCP middleware which provides a set of generic product related services, which can be used by collaborative PLM tools and applications, such as product engineering and product portfolio management. The envisaged services include collaborative product design, event management, the generation of product structures, and transparent access to external product models stored in CAD/PDM systems. These services access metadata describing product-related artifacts through a decentral metadata repository, which maintains these metadata in a decentral, virtualized manner via a P2P resource management layer. In the following, we discuss the different layers of the architecture.

**Fig. 3.** PCP top-level architecture

**Decentral Resource Management** The Siemens Resource Management Framework (RMF) [7, 6] provides decentral resource management based on a P2P
A peer to peer based approach to collaborative product engineering

RMF offers message-based communication between all nodes in the P2P network, fault tolerant publishing, discovery and retrieval of XML resources and an additional publish/subscribe mechanism. RMF implements a Distributed Hash Table - referred to as the P2P information space - based on the Chord algorithm [12] that is collaboratively created using the aggregated resources of all peers in the network. Queries for XML resources stored in the RMF information space can be specified using XPath expressions. The set of XML resources on which such a query is applied is determined by a set of keywords associated with every resource. In our application context, resources are metadata describing PLM-related artifacts, such as product models.

PCP Middleware

The middleware provides core and supporting operations on product models to form a decentral organised metadata model repository. Using operations we are able to define generic product related services. The first class of services is collaborative product design. These services support the handling of product models in the model repository (see e.g., use cases 2, 6, and 9 of Figure 2). New product models are formed and then published to the underlying network structure (use cases 3 and 7). Following the application scenario in section 3.1 the model is defined using product collaboration tools and applications (use case 3) before suppliers can add their proposals (use case 7).

Once a new model is available, product-related event management services are used to notify users. The resource management layer offers a subscription service to notify interested users (see Section 5 and use cases 4 and 8 at Figure 2). This notification is interpreted by middleware methods and forwarded to the relevant participants (use case 9). Metadata can be modified; this allows the incremental generation of product structures based on input of different participants (e.g., use case 5). In addition, the repository offers a search function (see section 5) to find model metadata (and thus, the actual models). Further methods include synchronization, versioning, and history build up. To synchronize the information space, list resources are formed and handled through the RMF (see Section 4 – “list resource”). The list resource contains e.g. versioning information of product models. Access to the actual product models stored in PDM or CAD systems is provided by a product repository access services. Our current system supports only the connection of STEP 3D-CAD-Files into product models. Middleware methods access corresponding user file systems on demand.

In the current implementation, the middleware services are accessed via a Graphical front-end implemented as an Eclipse plug-in (see Section 6).

4 Product Model Metadata Structure

In our approach, the P2P resource management maintains product model metadata. In this section, we describe the corresponding metadata structure.

Resources are the basic entities maintained by the RMF system. The back end is a resource. A resource is an XML element with RMF specific child elements. Two essential child elements are a unique ID as an identifier and a
list of keywords to enable a search for a published resource. We use two kind
of resources: part and list resources. To define a resource means that methods
implemented in the PCP middleware have to generate an XML document that
complies with the requirements of RMF resources. The following example shows
a part resource “‘part.xml’”: a model of a 2x2 lego brick.

```xml
<?xml version="1.0" encoding="ISO-8859-1"?>
<pa:part
xmlns:reg="http://siemens.com/rmf/registrar/v0_1"
xmlns:pa="http://patrick2.in.tu-clausthal.de/RMF/part"
xmlns:attr="http://patrick2.in.tu-clausthal.de/RMF/part_attr">
  <reg:id>0c2e18ac-3a33-4c3b-a20b-e87563194a2c</reg:id>
  <reg:keywords>
    <reg:keyword>139.174.100.169/3453</reg:keyword>
    <reg:keyword>Lego building brick</reg:keyword>
    <reg:keyword>request</reg:keyword>
  </reg:keywords>
  <pa:num>139.174.100.169/3453-1.0</pa:num>
  <pa:type>request</pa:type>
  <pa:character>component</pa:character>
  <pa:name>2x2 Lego building block</pa:name>
  <pa:description>Standard Lego building brick</pa:description>
  <pa:link>C:\MyCADProgram\Test.STEP</pa:link>
  <pa:part_attribute_1>
    <attr:number>1</attr:number>
    <attr:importance>mandatory</attr:importance>
    <attr:description>height</attr:description>
    <attr:unit>mm</attr:unit>
    <attr:value_min_request>10</attr:value_min_request>
    <attr:value_max_request>12</attr:value_max_request>
    <attr:value_real_request>-</attr:value_real_request>
    <attr:value_min_proposal>null</attr:value_min_proposal>
    <attr:value_max_proposal>null</attr:value_max_proposal>
    <attr:value_real_proposal>-</attr:value_real_proposal>
  </pa:part_attribute_1>
</pa:part>
```

Given the part resource in an XML file named “‘part.xml’”, it is possible
to instantiate a resource object as follows:

```java
SDOMSerializer serializer = SDOMFactory.getInstance().getDefaultSerializer();
SElement element = serializer.fromStream(new FileInputStream("part.xml"));
Resource resource = new Resource(element);
```

**Part resource** The part resource implements a mapping from real part properties to the RMF metadata model. As described in Section 3.1 a part is an information object with related attributes. There are two part resource types: One for *part requests* (first phase) and the other for *proposals* (second phase). Both are handled in one resource.

In the above example, we can see the most important metadata properties. A *part number* (example: ”139.174.100.169/3453-1.0”) consists of an IP address (139.174.100.169) followed by an in-house part identification (3435), its version (1) and variant number (0). The type classifies the resource, e.g., in *request for proposal* which must precede a *proposal*. The *structure* tag specifies if the part is an atomic ”component” or an ”assembly” composed of of multiple components. The last required value is a *link* to a local STEP-File (“‘C:\MyCADProgram\TEST.STEP’”) containing conceptual 3D data associated to a request or final 3D data describing a given proposal.
Each *part resource* consists of up to ten part attributes listing the importance, description, minimum, maximum and actual value of the attribute. The importance classifies the selected attribute as mandatory (example: height, length, width) or optional (example: color).

**List resource** A list resource contains all existing versions and variants of a part. It represents a central document to register changes on components or assemblies administrated through the middleware. On every system change, this document is consulted for the latest version and variant using the back end RMF Information Space. This method is necessary because of the possibility that two or more users will create a new part version/variant at the same time: We could expect an inconsistency. This can be avoided by using the list resource.

## 5 The Decentral Product Model Metadata Repository

In the previous section, we have defined how product model components (parts) are represented as resources. Now we show how these resources are stored and managed in a Decentral Product Model Metadata Repository, addressing the requirements in Section 2. The product structure is kept in a hierarchically organised part model navigation tree at each participant. Document management functions are realised by the ability of connecting files (e.g., CAD files) to product resources and display them. In future research, we shall group similar parts on the aim of reuse and integrate additional viewers to display advanced product structures based on their usage. In this section we describe the core functions provided by the repository.

**Registry** RMF provides a *Registrar* service to manage resources in the Information Space. If a user creates a new request for proposal (e.g., a request for a 2x2 Lego brick model) as shown on the left side of Figure 4, the registrar has to be initialised first. Then the part resource is published:

```java
registrar = rmf.getStrictRegistrar();
registrar.publish(resource);
```

Simultaneously a list resource is published, too. Later registered variants and versions are treated as new part proposal resources and have to be published separately. The list resource is only modified thereafter.

**Search** The first step in searching for a part (resource) is to determine the peer that hosts it. A user needs only to know a keyword of the resource he wants to retrieve, this may be a requested part for example. Afterwards all returned (request-) resources from the responsible peer are filtered using an XPath search query, so that a request can result in only one concrete part resource if required. The following example shows a search for one specific Lego brick model with the part number “139.174.100.169/3453”, version 1 and variant 0. This case occurs when selecting a specific part from the repository to display its details:
String keyword = "Lego building brick";
String num = "/part:part[part:num = "139.174.100.169/3453-1.0"]
try {
    results = m.registrar.search(keyword, num,
    new SNamespace[] {
        Part.PART_NAMESPACE,
        Part_attributes.PART_ATTRIBUTES_NAMESPACE,
        Part_list.PART_LIST_NAMESPACE
    });
} catch (Exception e){...}

**Modification** Resources that have been previously registered can be modified. This function is used, when a user reacts on a request and wants to publish a specific proposal. This means first the creation of a new part variant (or even a new version that has to be registered by modifying the list resource). Only if the modification process succeeds, a new part proposal can be registered in the Information Space. The new part resource contains values for the proposal fields, like “value-min-proposal” for example.

**Subscription** By subscribing for certain resources (based on keywords), participants are able to handle communication events on publication of new models in the Information Space. All participating peers subscribe for the keywords
“request” and “proposal”. Whenever a resource that matches one of the given keywords is added, modified, or removed, an event will be triggered to them.

6 Prototypical Implementation of a Metadata Repository Browsing and Editing Tool

As proof-of-concept and vehicle for testing the PCP system, we did a prototypical implementation of a front-end tool for accessing the Decentral Product Model Metadata Repository. The tool has been built as a plug-in using the Eclipse platform. Figure 5 shows a screenshot of the tool. The screenshot illustrates three views which were so far implemented: (1) a part repository view to display requests and proposals hierarchically ordered by enterprise / departments / model developers involved. Dialogs are available to add new requests and to search for existing models. (2) an editor for displaying details of selected product model requests. It offers the possibility for generate proposals. You can add new attributes or connect the proposals 3D-CAD files. (3) the ability to access (i.e., physically download) and display a selected 3D-CAD file of a product model.

![Fig. 5. Screenshot of the metadata browsing and editing tool](image)

7 Conclusion and Outlook

The contribution of this paper is threefold. First, we made a case for the benefits of decentral management functions for collaborative, cross-enterprise PLM, illus-
A peer to peer based approach to collaborative product engineering

treated by an application scenario; second, we described the architecture of PCP, a decentral product collaboration platform adding decentral management functions to PDM solutions; third, we described a core component of the PCP: the Decentral Product Model Metadata Repository, and presented the prototype implementation of a repository browser and model editor tool. This paper describes work-in-progress, its results lay the foundation for our future work on a collaborative PLM platform; there are numerous extensions of and challenges to our work. First, this paper has focused on the architecture and the repository. The remaining components need to be specified more thoroughly and implemented; a second challenge is to integrate our work into an existing PLM solution. We started a cooperation with Agile Corporation to integrate our concept with the Agile PLM solution. We hope to gain new insights from this and find collaborative use cases that are not yet handled in our architecture and will create new requirements. This will also allow us a more systematic evaluation of our work, far beyond the proof-of-concept implementation discussed in this paper.

References


3 http://www.agile.com/