Intelligente Produktkonfiguratoren als Voraussetzung für maßgeschneiderte Massenprodukte

A. Felfernig, G. Friedrich, D. Jannach[1]

Die Erzeugung variantenreicher Produkte stellt neue Anforderungen an betriebliche Prozesse im Zusammenhang mit der Produktion (Vertrieb, Produktionsplanung usw.). Konfigurationssysteme können zur Unterstützung dieser Prozesse einen wesentlichen Beitrag leisten. Voraussetzung für den Einsatz von Konfiguratoren ist eine möglichst schnelle Entwicklung, die eine Grundlage für einen effektiven Einsatz im betrieblichen Umfeld darstellt. Konfigurierbare Produkte besitzen komplexe Produktstrukturen und Abhängigkeiten zwischen einzelnen Bauteilen, die im Rahmen der Entwicklung und des Einsatzes eines Konfigurationssystems geeignet modelliert und gewartet werden müssen. Auf Grund zahlreicher Berührungspunkte zu anderen betrieblichen Prozessen muss ein Konfigurationssystem in die gesamtbetriebliche Datenverarbeitung eingebettet werden.

Dieser Beitrag präsentiert eine entsprechende Architektur für Konfigurationssysteme, welche die Entwicklungsphase durch graphische Modellierung und wissensbasierte Problemrepräsentation und den Konfigurationsvorgang mittels einer automatisch generierten graphischen Benutzeroberfläche beschleunigt und verbessert. Dies ermöglicht eine entsprechende Entwicklungszeitverkürzung, eine damit einhergehende Senkung der Entwicklungskosten und eine verbesserte Wartbarkeit für Konfigurationssysteme.

Schlüsselwörter: Produktkonfiguration; prototypische Anwendungsentwicklung; automatisierte Softwareentwicklung; Konfigurationssysteme; mass-customization

Intelligent configuration systems as a prerequisite for mass-customized products.

The production of products with many variants imposes new requirements on business processes in connection with sales, distribution, and logistics. Configuration systems can support these processes in an efficient way. The rapid development of these systems is a prerequisite for a successful deployment within the organizational context. Configurable products have complex product structures and there are dependencies between components, which have to be modeled and maintained during the development and deployment of the system. Additionally, configuration systems have to be integrated into the existing software-infrastructure of the organization.

This paper shows an architecture for configuration systems addressing the requirements mentioned above. The development of the system is supported by a graphical modeling language allowing for a simple knowledge representation. The interactive configuration procedure is enhanced through a graphical interface, which can be partly generated from the specification. The presented architecture allows for decreased development time and costs and increased maintainability for configuration systems.

Keywords: product configuration; prototyping; automated software development; configuration systems; mass-customization

1. Einleitung

Das Design, die Produktion und der Vertrieb von hochvarianten, in starkem Maße den Kundenwünschen angepassten Produkten gewinnen durch die zunehmende Kundenorientierung vieler Unternehmen verstärkt an Bedeutung. Diese erhöhte Aufmerksamkeit diesem Thema gegenüber spiegelt sich einerseits im betriebswirtschaftlichen Kontext [8], andererseits auch im Bereich der Informatik-Forschung [1] wider. Immer

[1] Dipl.-Ing. Alexander Felfernig, O. Univ. Prof. Dipl.-Ing. Dr. Gerhard Friedrich, Dipl.-Ing. Dietmar Jannach, Institut für Informationstechnologie, Lehrstuhl für Produktionsinformatik, Universität Klagenfurt, Universitätsstraße 65-67, A-9020 Klagenfurt.

mehr Produkte werden in immer größerer Variantenanzahl unter dem Schlagwort mass-customization[2] [8] produziert, wobei die Produktlebenszeit und somit auch der Entwicklungszyklus zunehmend engeren Zeitschranken unterliegen.

Die Entscheidung, maßgeschneiderte Produkte herzustellen und zu vertreiben, betrifft viele Unternehmensbereiche, welche auf diese neue Art der Produktion erst umgestellt werden müssen. Dies beginnt beim Produktdesign, wo die konfigurierbaren Anteile des Produkts festgelegt werden müssen, reicht über Produktion und Logistik bis hin zu Vertrieb und Aufbau von technischen Anlagen.

Um mit den neuen Herausforderungen von Variantenprodukten umzugehen, werden im zunehmenden Maße Konfiguratoren entwickelt, die beim Vertrieb und der Produktion des Produkts unterstützen. Ein Konfigurationssystem (Konfigurator) ist ein Informationssystem, welches die Erstellung, Gültigkeitskontrolle und Administration von konfigurierbaren Produkten unterstützt. Auf Basis der Kundenwünsche erstellt der Konfigurator die für den Kunden passende Konfiguration, wobei alle Restriktionen, wie z.B. ein vorgegebener Höchstpreis, eingehalten werden. Die zunehmende Bedeutung von Konfigurationssystemen spiegelt sich nicht zuletzt in Erweiterungen zur Bewältigung von Variantenprodukten in betriebswirtschaftlicher Standardsoftware wider.

Konfigurationsaufgaben haben durch ihre typische Struktur und Art der Problemlösung die Verwendung von Systemen der künstlichen Intelligenz hervorgerufen und verschiedene Ansätze hervorgebracht. Diese Konfigurationsmethoden leiden jedoch an verschiedenen Mängeln, welche aus schwieriger Erstellung der Wissensbasis, damit verbundener schlechter Wartbarkeit und mangelnder Integration in die übrige DV-Landschaft resultieren.

In dieser Arbeit werden zuerst Einsatzmöglichkeiten von Konfigurationssystemen im betrieblichen Umfeld und die damit verbundenen Probleme erläutert. Wir präsentieren eine Entwicklungsumgebung für Produktkonfiguratoren, welche auf einer allgemeinen und leicht verständlichen Modellierungssprache basiert und die Entwicklungszeiten für Konfigurationssysteme verkürzt und eine Kostensenkung ermöglicht.

2. Existierende Ansätze und Einsatzmöglichkeiten für Konfiguratoren

Es existiert eine Vielzahl von formalen Definitionen einer Konfigurationsaufgabe. Informell kann Konfiguration als eine Designaktivität gesehen werden, wobei folgende Eigenschaften als typisch gelten [6,9]:

- Das konfigurierte Produkt besteht aus einer Menge von Bauteilen von vordefinierten Komponententypen und ihren Parametern.
- Komponenten können mit anderen in vordefinierter Weise in Beziehung stehen.

Das Ergebnis des Konfigurationsvorgangs ist eine Auswahl von Komponenten und eine Belegung der Parameter, wobei sowohl die technischen Restriktionen als auch die Kundenanforderungen erfüllt sind. Neben dieser Liste von Bauteilen ist auch die Topologie, d.h. die Verbindungen zwischen den Bauteilen im konfigurierten Produkt, von Bedeutung.

Abhängig von einer Menge von technischen und betriebswirtschaftlichen Anforderungen werden Konfiguratoren in unterschiedlichen Anwendungskontexten eingesetzt. Typische Bereiche sind [10]:

- Vollständig automatisierte Konfiguration: Sie ermöglicht die effektive Unterstützung von Benutzern, die über kein tiefes Hintergrundwissen aus dem Anwendungsbereich verfügen. Technische Details werden durch Default-Einstellungen bzw. nach Standardvorgaben konfiguriert und müssen nicht interaktiv bestimmt werden.
- Verifikation von interaktiv entwickelten Lösungen: Hier dient der Konfigurator der Überprüfung der Korrektheit von interaktiv gestalteten Konfigurationen.
- Unterstützung der Vertriebsabteilung bei der Angebotserstellung: Dem Kunden werden bestimmte Auswahlmöglichkeit angeboten, die anschließend von der technischen Konfiguration zu einer vollständigen Produktspezifikation umgesetzt werden (Sales Configurator).
- Technische Aufbauplanung: Werden durch den Konfigurationsvorgang auch Struktur- bzw. Verbindungsmerkmale ermittelt, so kann das Konfigurationsergebnis auch als Anleitung für die technische Aufbauplanung verwendet werden.

[2] Herstellung von in hohem Maße den Kundenwünschen angepassten Gütern zu Massenfabrikationspreisen (auch: Post-Mass-Production – Paradigm).

3. Architektur integrierter Konfigurationssysteme

Ein Konfigurationssystem kann nicht für sich alleine stehen, sondern existiert auf Grund der Einflüsse auf andere Unternehmensbereiche immer im Zusammenhang zum übrigen betrieblichen Datenverarbeitungsumfeld. Abb. 1 stellt die Einbettung eines Konfigurationssystems in die betriebliche Informationsverarbeitung dar.

- Produktmodellierung/Produktmodell: Mit Hilfe einer Wissensakquisitionskomponente wird ein generisches Produktmodell erstellt und anschließend formalisiert in die Wissensbasis übernommen. Die verwendbaren Bauteile werden der Materialwirtschaftskomponente des Gesamtsystems entnommen. Die Anforderungen an das Produkt entstehen einerseits aus den Anforderungen des Marketing und Produktmanagement, andererseits aus technischen Beschränkungen und Möglichkeiten aus der Produktion.
- Konfigurator/Solver: Ein Inferenzmotor erzeugt auf Basis der Wissensbasis (Produktmodell) Konfigurationen, welche sich im wesentlichen aus technischen Stücklisten, technischer Aufbaubeschreibung und möglicherweise Arbeitsplänen zusammensetzen. Diese Konfigurationsergebnisse werden an die EDV der anderen betroffenen Betriebsbereiche übergeben. Die notwendigen Bauteile verursachen zum Beispiel einen Teilebedarf und einen Beschaffungsvorgang in der Beschaffungslogistik.
- Graphische Oberfläche/ Interaktive Konfiguration: Während des interaktiven Konfigurationsvorgangs müssen bei Erfassung der Kundenwünsche sowohl die technischen Restriktionen aus dem Produktmodell als auch produktionslogistische Aspekte geprüft werden. Zur Fertigstellung einer Konfiguration muss die Realisierbarkeit mit den vorhandenen Ressourcen aus Produktionsplanung und Materialwirtschaft sowie die Einhaltung von Terminen gewährleistet sein.

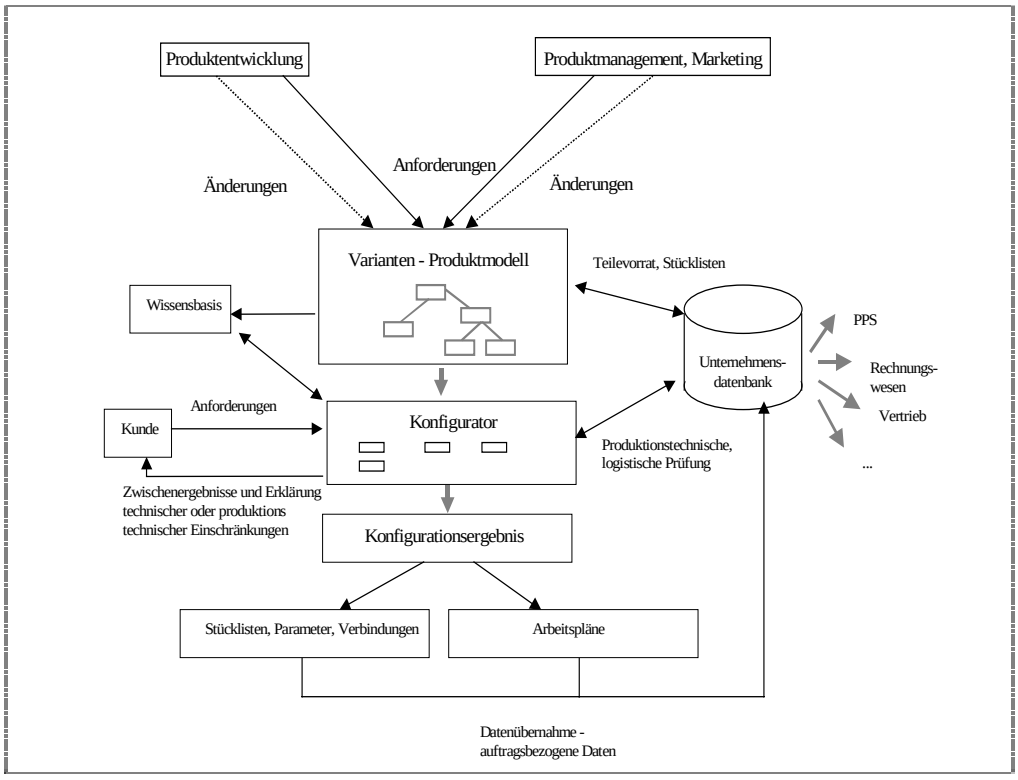

Abb. 1. Produktkonfiguration im betrieblichen Umfeld

Bei Integration in die übrige betriebliche Datenverarbeitung ergeben sich durch den Einsatz von Konfigurationssystemen folgende Vorteile:

- Rasche Reaktion auf Kundenanfragen: Der Konfigurator überprüft auf Basis der aktuellen Kundenanforderungen sofort die Machbarkeit (vertriebsbezogen, technisch, produktionstechnisch usw.) und gibt entsprechende Informationen.
- Vermeidung ungültiger Bestellungen: Nicht realisierbare Produktanforderungen werden sofort durch oben genannte Machbarkeitsuntersuchungen entdeckt.

- Beschleunigung des Verkaufszyklus: Nicht realisierbare Bestellungen werden vermieden, so kommt es zu keinen unnötigen Rekonfigurationen, die eine entsprechende Verzögerung der Auftragsbearbeitung verursachen.
- Schnellere Anpassung: Auf sich ändernde Märkte und Kundenbedürfnisse erfolgt im Rahmen der technischen Möglichkeiten eine flexible Anpassung. Die sonst entstehende Lücke zwischen technischer Realisierbarkeit und organisatorischer Flexibilität wird wesentlich kleiner.
- Steigerung der Qualität: Komplexe (technische) Produktstrukturen erfordern entsprechende Repräsentations- und Verarbeitungsmechanismen, um ein fehlerfreies Produkt zu fertigen und auszuliefern.
- Unterstützung anderer Unternehmensbereiche: Konfiguratoren unterstützen die Lösung von Problemen in Unternehmensbereichen, die von der mass-customization betroffen sind. Beispielsweise verringert sich der Schulungsaufwand für Vertriebsmitarbeiter, da das notwendige Detailwissen vom Konfigurator zur Verfügung gestellt wird.

4. Verbesserungen in Konfigurationssystemen

Vorhandene Produktkonfiguratoren bringen erhebliche Einsparungen, wie es z.B. die Anwendung des Konfigurators COCOS (Configuration by Constraint Satisfaction) im Bereich der Projektierung von EWSD-Anlagen (Elektronisches Wählsystem Digital) gezeigt hat [1]. Vor allem die durchgängige Unterstützung des Produktionsprozesses bringt erhebliche Kosteneinsparungen. Trotz aller bereits erreichten Verbesserungen und Vorteile sind die Entwicklung und der Einsatz von Konfiguratoren immer noch mit einigen Herausforderungen verbunden. In den folgenden Abschnitten wird auf diese Herausforderungen eingegangen.

4.1 Produktmodellierung

4.1.1 Problemstellung

Die klassischen Ansätze zur Modellierung von Variantenprodukten, wie z.B. Gleichteilestücklisten oder Plus-Minus-Stückliste, reichen in ihrer semantischen Ausdrucksmächtigkeit für die Modellierung von komplexen Produktstrukturen und Abhängigkeiten hochvarianter Produkte nicht mehr aus. Probleme treten z.B. dann auf, wenn es darum geht, Beziehungen zwischen einzelnen Komponenten zu formulieren, wie z.B. ‚Komponente X schließt Komponente Y aus' oder ‚Komponente X erfordert gleichzeitig die Installation von Komponente Y' usw. Solche Einschränkungen werden in Konfiguratoren mittels Regeln oder Constraints implementiert, haben aber keine Repräsentation in den entsprechenden Gleichteile- bzw. Plus-Minus-Stücklisten.

Wissensbasierte Produktkonfiguratoren verwenden für die Modellierung der möglichen Varianten und Beschränkungen zumeist eine proprietäre Modellierungsmethode bzw. Modellierungssprache. Diese Art der Wissensrepräsentation ist oft schwer verständlich und schwer zwischen den verschiedenen an der Erstellung beteiligten Personen kommunizierbar. Die Durchführung von Änderungen in einer umfangreichen Wissensbasis bringt Wartungsprobleme mit sich, zumal Abhängigkeiten zwischen Regeln nur schwer erkannt werden können [5].

Die korrekte Erstellung und Wartung ist nur mit Hilfe eines Knowledge-Engineers möglich. Um es z.B. dem technischen Fachexperten zu ermöglichen, selbständig die Produktstruktur zu definieren, sind verständliche, allgemein bekannte Modellierungskonzepte notwendig. Unsere Erfahrung zeigt, daß UML - Unified Modeling Language [2] - auch von einem Fachexperten ohne spezielle Informatikkenntnisse zur Modellierung des Konfigurationswissens eingesetzt werden kann. Ein wesentlicher Vorteil dieses Ansatzes besteht in der Geringhaltung der „Fehlerquelle" Knowledge-Engineer, der das Wissen des Fachbereichsexperten unter Umständen verfälscht oder unvollständig formalisiert.

4.1.2 Verwendung objektorientierter Modellierungswerkzeuge

Ein Ansatz zur Problemlösung besteht in der Verwendung von objektorientierten Modellierungskonzepten zur Modellierung des Konfigurationswissens [7]. UML ermöglicht die Definition von Stereotypen zur Realisierung domainspezifischer Modellierungskonzepte. Auf Basis dieses Stereotypkonzepts definiert unser Ansatz ein Modellierungssystem für die Modellierung von Konfigurationswissen, dessen Semantik oben angegebene Beschränkungen bezüglich der Ausdrucksmächtigkeit aufhebt.

Folgende Modellierungskonzepte haben sich als typisch für Konfigurationsprobleme erwiesen:

- Komponenten beschreiben vordefinierte Bauteile, welche durch Attribute charakterisiert werden.
- Ähnliche Komponenten können in einer Subtypisierungs- bzw. Generalisierungshierarchie stehen.
- Komponenten bestehen wieder aus anderen Komponenten, was durch Aggregation, verbunden mit den Ober- und Untergrenzen (Kardinalitäten) in UML ausgedrückt werden kann.

- Vor allem elektronische Produkte können über vordefinierte Verbindungen (Steckplätze), Ports, verbunden werden, welche auch als Anleitung für den Zusammenbau verwendet werden können.
- Der Einbau von einzelnen Komponenten kann den Einbau einer anderen Komponente erfordern (requires-Verbindung) bzw. ausschließen (incompatible-Verbindung).
- Ressourcen sind Funktionalitäten oder Eigenschaften des fertigen Produkts, welche von einzelnen Komponenten zur Verfügung gestellt bzw. verbraucht werden können. Diese Ressourcen müssen im fertigen Produkt balanciert sein.
- Spezielle Beschränkungen, welche graphisch nicht ausdrückbar sind, können unter Angabe von OCL-Constraints (Object Constraint Language) angegeben werden. OCL ist eine standardisierte Sprache im Rahmen von UML.

Abb. 2 zeigt ein generisches Produktmodell eines konfigurierbaren PCs, welches die oben beschriebenen Modellierungskonzepte verwendet und die Darstellungsform demonstriert.

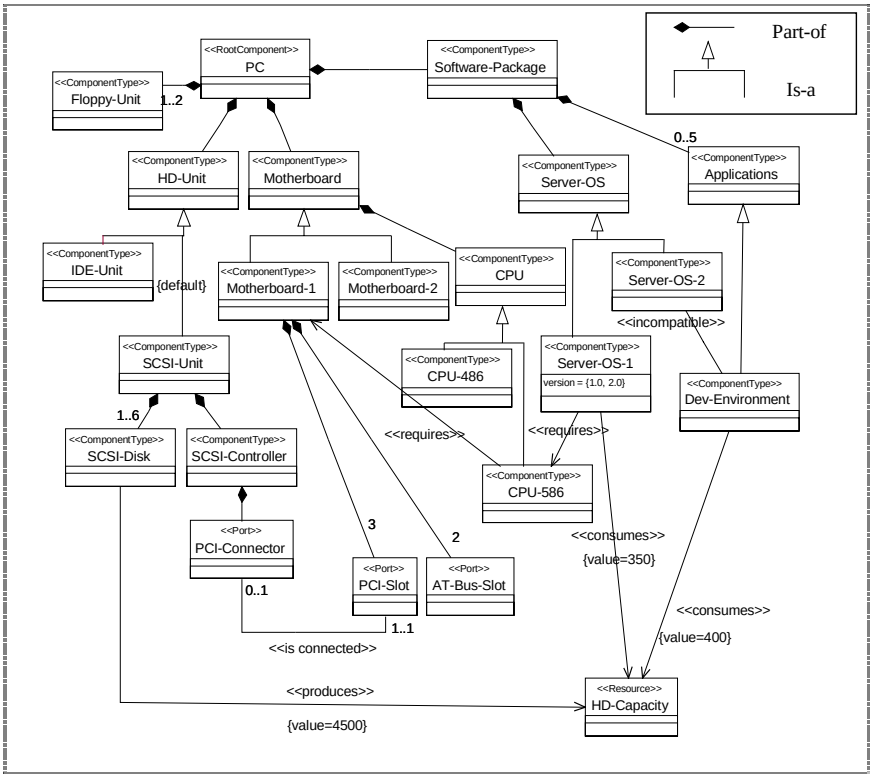

Abb. 2: Produktmodell eines konfigurierbaren PC

4.2 Entwicklungszeiten und automatisierte Konfiguatorentwicklung

4.2.1 Problemstellung

Die für die Entwicklung des Konfigurators zur Verfügung stehende Zeit ist kurz und begrenzt. Kurze Produktlebenszyklen erfordern eine parallele Entwicklung des konfigurierbaren Produkts und des Konfigurationssystems. Wenn ein neues (hochvariantes) Produkt auf den Markt gebracht wird, muss sofort ein entsprechender Konfigurator zur Unterstützung von Vertrieb, Produktion usw. zur Verfügung stehen. Änderungen im Produkt (z.B. neue Varianten) müssen leicht im Konfigurator nachziehbar sein.

4.2.2 Automatisierte Generierung von Wissensbasen

Unser Ansatz ermöglicht eine (teil-)automatisierte Generierung einer weiterverarbeitbaren Wissensbasis aus dem in der Wissensakquisitionsphase erstellten konzeptuellen Konfigurationsmodell.

Wir definieren ein logisches Modell, welches festlegt, was unter einem Konfigurationsproblem bzw. der Lösung eines solchen zu verstehen ist. Dieses Modell legt die Struktur der Wissensbasis fest. Grundlage ist das Component-Port-Modell, welches sich für die Lösung beliebiger Konfigurationsprobleme eignet [6] und als Basis für allgemeine Konfigurationslöser [1,11] dient.

Das Domänenwissen (verwendbare Komponenten und Einschränkungen) sowie die Anforderungen der Benutzer werden durch logische Sätze (range restricted clauses) beschrieben. Das Ergebnis einer Konfiguration besteht aus einer Menge von Fakten (verwendete Komponenten und Verbindungen zwischen den Komponenten), wobei eine gültige Konfiguration konsistent mit dem Domänenwissen und den Benutzeranforderungen sein muss.

Für alle Modellierungskonstrukte im konzeptuellen UML-Modell (z.B. Aggregation, Generalisierung) wird festgelegt, wie diese in die Form von logischen Sätzen transformiert werden müssen. So wird das graphische Modell schrittweise in eine verarbeitbare Form gebracht und eine präzise Semantik für alle Modellierungskonzepte definiert.

Durch die automatische Generierung der verarbeitbaren Wissensbasis wird die Erstellung von Prototypen stark erleichtert. Der Wissenserwerb wird ausschließlich auf der Ebene der Produktmodellierung mittels UML durchgeführt; der technische Experte muss sein Wissen nicht in prädikatenlogischer Form formulieren, sondern verwendet ein verständliches und kommunizierbares Werkzeug[3].

Bei Änderungen der Produktstruktur muss das generische Produktmodell verändert werden und in einem weiteren Übersetzungslauf eine neue Wissensbasis erzeugt bzw. die alte erweitert werden. Eine detaillierte Darstellung der logischen Theorie sowie die Regeln zur Ableitung der Regeln für diese Theorie kann in [3] gefunden werden.

4.3 Erstellung grafischer Benutzeroberflächen

4.3.1 Problemstellung

Viele Konfiguratoren sind auf bestimmte Anwendungsdomänen zugeschnitten, speziell die graphische Benutzeroberfläche lehnt sich stark an die Gegebenheiten der betrachteten Domäne an. Soll nun ein Konfigurator in einer ähnlichen Domäne eingesetzt werden, erfordert dies im einfachsten Fall die Implementierung bzw. Anpassung der Benutzerschnittstelle. Zur Vermeidung einer neuerlichen Implementierung der graphischen Oberfläche muss ein Mechanismus zur (teil-)automatisierten Generierung derselben gefunden werden.

4.3.2 Extraktion von Benutzerentscheidungen, Konfigurationssteuerung

Ausgehend vom generischen Produktmodell kann eine automatische Erstellung solcher Benutzerschnittstellen unterstützt werden. Durch die Analyse der generischen Produktstruktur können folgende Entscheidungsmöglichkeiten für den Benutzer extrahiert werden (sofern diese nicht durch Beschränkungen bereits festgelegt sind):

- Die Verwendung einer Generalisierung bringt die Auswahl eines zu verwendenden Subtypen mit sich.
- Eine unbestimmte Multiplizität einer Aggregation muss für ein fertiges Produkt spezifiziert werden.
- Unbestimmte Attribute von Komponenten müssen gesetzt werden, wenn sie nicht durch andere Beschränkungen und Auswahlen schon vorbestimmt sind.
- Bestimmte Ressourcenanforderungen (wie z.B. ein maximaler Preis) können durch den Benutzer festgelegt werden.
- Optionale Verbindungen können geschlossen werden.

Weitere Aspekte für die Erstellung von Konfigurationsoberflächen sind:

- Die Vorgabe von Schlüssel-Komponenten muss möglich sein.
- Für jede einzelne erzeugte Instanz eines Komponententyps müssen separat die Attribute und Verbindungen eingestellt werden können.

Nach Extraktion der Benutzerentscheidungen ist es möglich, eine einfache Oberfläche für den interaktiven Konfigurationsvorgang zu erzeugen. In unserem Ansatz wird das Konfigurationsergebnis in Tabellen eines relationalen Datenbanksystems abgelegt. Die Ablegung der Daten in Tabellen ist sinnvoll, da die Verwendung von relationalen Tabellen leicht verständlich und verbreitet ist sowie eine Integration in die übrige betriebliche DV-Landschaft ermöglicht.

[3] Verwendet wird nur die Statikmodellierung von UML, welche um Stereotypen für Konfigurationsaufgaben erweitert wird.

4.3.3 GUI-Builder

Es existieren verbreitete Werkzeuge für die Erstellung von Benutzeroberflächen, welche auf relationalen Tabellen basieren (Oracle Forms, MS Access usw.). Diese Werkzeuge erlauben schnelle und prototypische Erstellung von Anwendungen und vermindern den nötigen Programmieraufwand.

4.3.4 Web-Applikationen

Der Vorteil dieser webbasierten Konfiguratoren liegt in der leichten „Transportierbarkeit“ durch den Vertriebsmitarbeiter. Für einfache Konfigurationsprobleme kann der Kunde selbst die Konfiguration durchführen und über das Internet einen Bestellauftrag erzeugen. Dies ermöglicht es, auch konfigurierbare Produkte elektronisch anzubieten und zu konfigurieren. Bei Verwendung von Konfiguratoren, welche vom restlichen System (kurzzeitig) losgelöst sind, entsteht bei Lösen der Verbindung das Problem der Inkonsistenz von Datenbeständen, da die Daten des PCs des Vertriebsmitarbeiters nicht aktualisiert werden können. Bei webbasierten Applikationen operiert der Benutzer immer auf den aktuellen Datenbeständen.

Die Schwierigkeit bei webbasierten Applikationen liegt in der Erstellung der Ein- und Ausgabemasken. Durch die Verwendung von neuen Werkzeugen und verbreiteten Scriptsprachen (JavaScript, VisualBasic-Script in ASP - Active Server Pages) kann aber auch der Zugriff auf Daten einer relationalen Datenbank relativ leicht erfolgen.

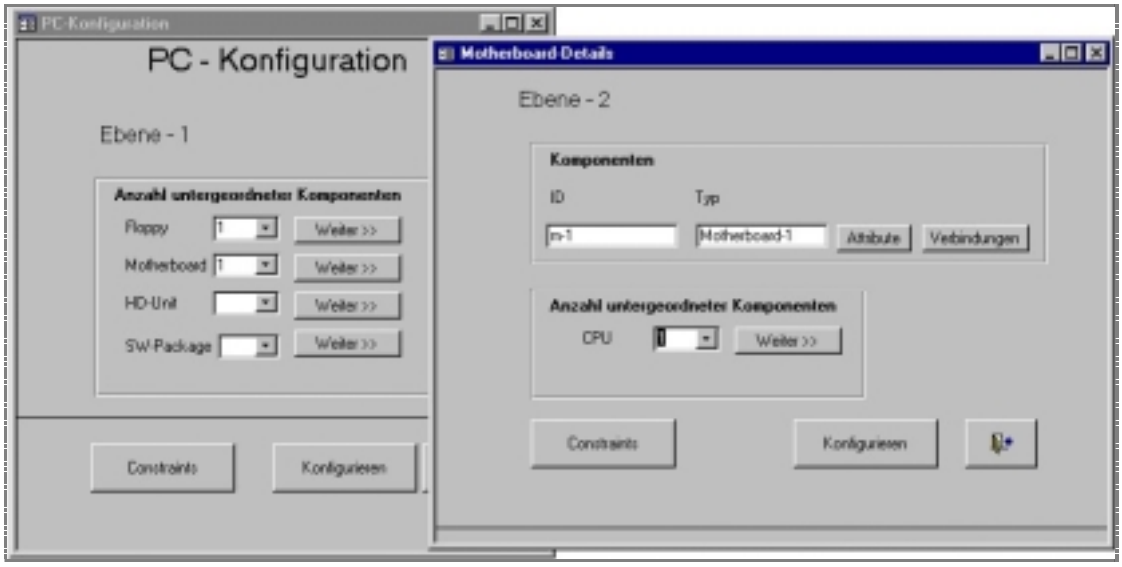

Abb. 3: Beispiel für eine generierte Konfigurationsoberfläche

Abb. 3 zeigt eine beispielshafte generierte Oberfläche. Es wird ein top-down-Ansatz verfolgt, wobei bei der Wurzel der generischen Produktstruktur mit der Konfiguration begonnen wird. Für alle untergeordneten Bauteile wird ein Auswahlschalter definiert, welcher die Anzahl der untergeordneten Bauteile festlegen lässt. Mit Hilfe einer Schaltfläche (Weiter) kann ein bestimmter Bauteil bzw. eine Baugruppe (Komponente) weiter spezifiziert werden.

5. Zusammenfassung und Ausblick

In die betriebliche Landschaft eingebettete Konfigurationssysteme ermöglichen eine effiziente Unterstützung betrieblicher Prozesse in Produktion und Vertrieb von variantenreichen Produkten. Neben der Darstellung von existierenden Ansätzen, Einsatzmöglichkeiten und Vorteilen von Konfiguratoren bringt unser Beitrag Vorschläge für die Verbesserung und Beschleunigung bei der Entwicklung und Wartung solcher Systeme. Die Beschreibung einer generischen Produktstruktur bzw. die Formalisierung in einer Wissensbasis sind entscheidende Punkte. Sie werden durch eine graphische, verständliche und allgemein einsetzbare Modellierungsmethode in geeigneter Weise unterstützt. Aus dem generischen Produktmodell können automatisch graphische Konfigurationsoberflächen erzeugt und Entwicklungszeiten weiter reduziert werden.

Konfiguratoren allein sind noch keine Garanten zur Nutzung vorhandener Einsparungspotentiale. Wesentliche Voraussetzung dafür ist die Einbettung der Konfiguration in betroffene betriebliche Prozesse, um nicht einzelne Aufgaben sondern den ganzen Prozess zu unterstützen.

Methoden objektorientierter Modellierung bieten die Möglichkeit, das Konfigurationswissen in geeigneter Art und Weise darzustellen, um es in weiterer Folge leicht in eine Wissensbasis zu transformieren. So erfolgt eine Integration von Methoden des Software-Engineerings mit jenen der künstlichen Intelligenz mit dem Effekt, dass im Bereich des Wissenserwerbs Standard-Modellierungskonzepte zur Verfügung gestellt werden. In weiterer Folge können diese Modelle durch eindeutige Abbildungsvorschriften in eine logikbasierte Darstellung mit klarer Semantik transformiert werden. Der Grad der Einsetzbarkeit und Akzeptanz von wissensbasierten Konfiguratoren kann so wesentlich gesteigert werden.

6. Schrifttum

[1] Fleischanderl, G., Friedrich, G., Haselböck, A., Schreiner, H., Stumptner, M.: Configuring large systems using generative constraint satisfaction. IEEE Intelligent Systems, (Juli/August 1998).

[2] Fowler, M., Scott, K.: UML distilled – Applying the standard object modeling language. Addison Wesley, (1997).

[3] Friedrich, G., Stumptner, M.: Consistency-based configuration. University of Klagenfurt, Technical Report KLU-IFI-98-5, (1998).

[4] Heinrich, M., Jüngst, E.W.: A resource-based paradigm for the configuration of technical systems from modular components. Proc. 7th IEEE Conference on AI Applications (CAIA), S. 257-264, (1991).

[5] McDermott, J.: R1: A rule-based configurer of computer systems. Artificial Intelligence, (1982).

[6] Mittal, S., Frayman, F.: Towards a generic model of configuration tasks. Proc. Eleventh Int. Joint Conf. Artificial Intelligence, S. 1395-1401, (1989).

[7] Peltonen, H., Männistö, T., Alho, K., Sulonen, R.: Product configurations - an application for prototype object approach, Tokoro, Pareschi (eds.): Object oriented programming. 8th European Conference, ECOOP'94, S. 513–534. Berlin Heidelberg New York: Springer, (1994).

[8] Pine II, B. J., Victor, B., Boynton, A. C.: Making mass customization work. Harvard Business Review, S. 108-119, (Sep.-Okt. 1993).

[9] Sabin, D., Weigel, R.: Product configuration frameworks. IEEE Intelligent Systems, (Juli/August 1998).

[10] Stumptner, M.:An overview of knowledge-based configuration. AI Communications 10, S. 111-126 (1997).

[11] Stumptner, M., Haselböck, A., Friedrich, G.: Cocos: A tool for constraint-based, dynamic configuration. Proc. CAIA '94: 10th Conf. AI for Applications, S. 373-380. Calif.: IEEE Computer Society Press, (1994).