Integrationswerkzeuge für verfahrenstechnische Entwicklungsprozesse

Simon M. Becker, Priv.-Doz. Dr. Bernhard Westfechtel
Lehrstuhl für Informatik III, RWTH Aachen
(sbecker|bernhard)@i3.informatik.rwth-aachen.de

1 Einleitung

Der Sonderforschungsbereich 476 IMPROVE [4], der 1997 an der RWTH Aachen eingerichtet wurde, befasst sich mit der informatischen Unterstützung übergreifender Entwicklungsprozesse in der Verfahrenstechnik. Arbeitsschwerpunkte liegen u.a. in der Modellierung verfahrenstechnischer Entwicklungsprozesse und ihrer Produkte sowie in der Entwicklung innovativer Werkzeuge. In diesem Beitrag berichten wir über Integrationswerkzeuge, mit deren Hilfe im Entwicklungsprozess entstehende Dokumente miteinander konsistent gehalten werden [2]. Diese Werkzeuge liefern somit einen Beitrag zur Datenintegration.

In verfahrenstechnischen Entwicklungsprozessen wird eine Vielzahl von unterschiedlichen Werkzeugen eingesetzt, mit denen die Entwickler verschiedene heterogene Dokumente bearbeiten, wie z.B. Fließbilder, Simulationsmodelle und Simulationsergebnisse. Diese Dokumente werden unabhängig voneinander in Dateien oder Datenbanken abgespeichert. Inhaltlich betrachtet, bestehen jedoch zahlreiche feingranulare Abhängigkeiten zwischen den einzelnen Dokumenten. So muss z.B. das Simulationsmodell konsistent zum zu simulierenden Fließbildausschnitt sein. Die Konsistenz zwischen den Dokumenten sicherzustellen, verursacht einen beträchtlichen Aufwand, insbesondere wenn sie von verschiedenen Entwicklern bearbeitet werden. Hinzu kommt, dass bei Concurrent und Simultaneous Engineering häufig inkrementelle Änderungen durch viele Dokumente propagiert werden müssen. Integrationswerkzeuge modellieren explizit die feingranularen Abhängigkeiten zwischen den Dokumenten und nutzen die gewonnene Information, um die Entwickler bei deren Abgleich zu unterstützen.

Insgesamt zeichnen sich die hier präsentierten Integrationswerkzeuge durch folgende Eigenschaften aus:

- Integrationswerkzeuge unterstützen den Benutzer durch dokumentübergreifende Konsistenzanalysen, Browsing-Funktionen und Transformationen.
- Sie arbeiten inkrementell, d.h. sie propagieren Änderungen in davon betroffene Dokumente, und unterscheiden sich dadurch von den industriell üblichen Konvertern, die vollständige Dokumente transformieren.
- Sie propagieren Änderungen in aller Regel bidirektional, d.h. z.B. je nach Bedarf vom Simulationsmodell zum Fließbild oder umgekehrt.
- Sie werden auf Anforderung aktiviert, um dem Benutzer die Entscheidung zu überlassen, wann eine Änderungspropagation sinnvoll ist.

- Sie operieren modellbasiert, d.h. sie beruhen auf Modellen der zu verknüpfenden Dokumente, die z.B. den Aufbau von Fließbildern oder Simulationsmodellen beschreiben.
- Sie werden durch Integrationsregeln gesteuert, mit denen festgelegt wird, welche Modellelemente zueinander in Beziehung gesetzt werden können und wie Muster solcher Elemente zueinander korrelieren.
- Sie arbeiten automatisch, wo dies möglich ist, und interaktiv, wenn die Integrationsregeln mehrdeutig sind. In diesem Fall entscheidet der Benutzer, welche Regel anzuwenden ist.
- Sie unterstützen die A-posteriori-Integration von Werkzeugen unterschiedlicher Hersteller.

Im Folgenden führen wir zunächst ein motivierendes Beispiel ein und befassen uns dann mit den Grundlagen der Modellierung, auf denen die Integrationswerkzeuge basieren. Danach erläutern wir die Integrationswerkzeuge selbst zunächst auf konzeptioneller Ebene und gehen dann auf deren Implementierung ein.

2 Szenario

Um den Nutzen inkrementeller Integrationswerkzeuge zu demonstrieren, präsentieren wir im Folgenden einen kleinen Beispielablauf, der die Integration von Fließbildern und Simulationsmodellen zeigt (Abbildung 1). Zwischen Fließbildern und Simulationsmodellen bestehen enge Beziehungen, aber es sind nicht notwendigerweise 1:1-Beziehungen. In aller Regel lassen sich Simulationsmodelle nicht automatisch aus Fließbildern ableiten oder umgekehrt. Schließlich müssen im Zuge des hochgradig iterativen Entwurfsprozesses Fließbilder und Simulationsmodelle wiederholt miteinander abgeglichen werden. Der Bedarf an intelligenter Werkzeugunterstützung ist hier besonders hoch.

Konkret nehmen wir an, dass das Fließbild in Comos PT [6] erstellt wird und die Simulation mit Aspen Plus [5] durchgeführt wird. Es sind also Werkzeuge unterschiedlicher Hersteller beteiligt, die a posteriori integriert werden. Für dieses Werkzeugpaar haben wir ein Integrationswerkzeug implementiert, das auf einem allgemeinen Rahmenwerk basiert (s. Abschnitt 5). Der folgende Beispielablauf zeigt auf konzeptioneller Ebene, in welcher Weise dieses Werkzeug den Abgleich von Comos-Fließbildern mit Aspen-Simulationsmodellen unterstützt.

Als verfahrenstechnischen Prozess betrachten wir die Produktion von Ethanol aus Ethen und Wasser. Der Beispielablauf umfasst folgende Schritte:

1. In Comos wird ein initiales Fließbild erstellt, das zunächst nur einen Teil des Prozesses beschreibt (nämlich das Erhitzen der Lösung und die Reaktion in einem Rohrreaktor).
2. Das initiale Fließbild wird mit Hilfe des Integrationswerkzeugs in ein Simulationsmodell in Aspen übertragen. Bei der Erzeugung des Simulationsmodells muss der Benutzer zwei Entscheidungen treffen. Der Erhitzungsschritt lässt sich zwar strukturell 1:1 ins Simulationsmodell übertragen, der Benutzer muss aber aus den verfügbaren Blöcken einen geeigneten für die Simulation auswählen. Beim Rohrreaktor, der in der Simulation auf

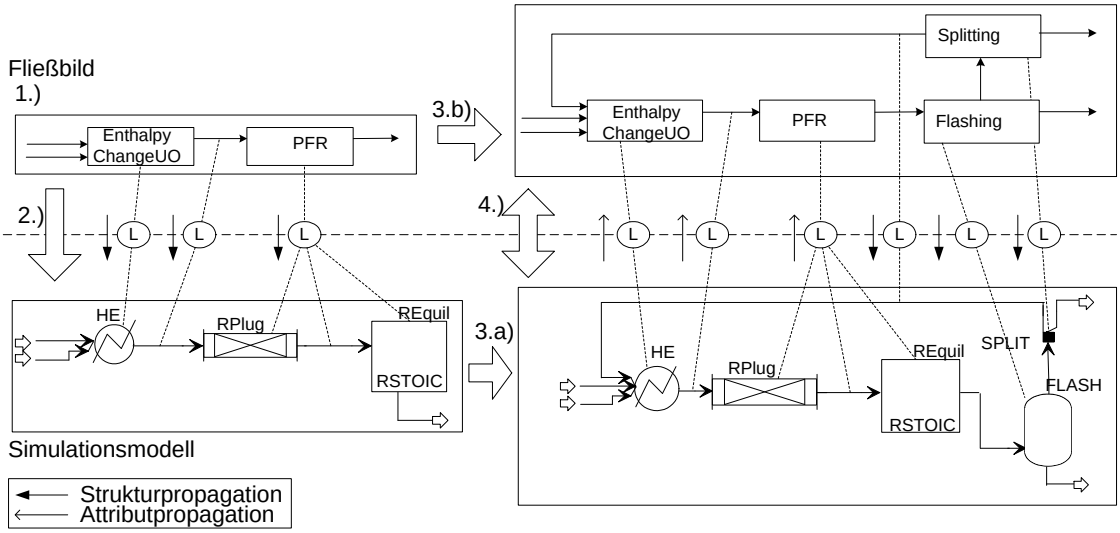


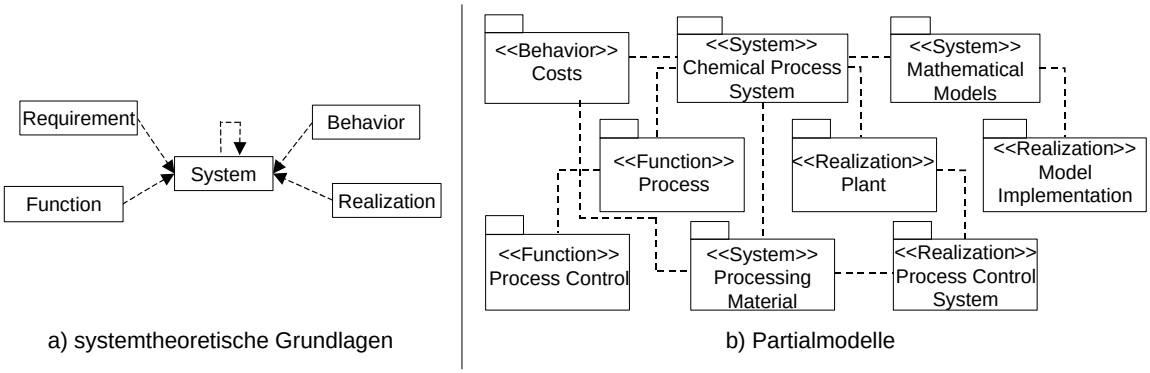
Abbildung 1: Anwendungsbeispiel: Integration von Fließbildern und Simulationsmodellen

mehrere Blöcke aufgeteilt werden muss (1:n-Beziehung), muss der Benutzer die entsprechende Abbildung als Alternative zu der Standardabbildung auf einen Reaktor in Aspen wählen.

3. Das vom Integrationswerkzeug erzeugte Simulationsmodell ist noch unvollständig. Es wird (in Aspen) um weitere Simulationsparameter ergänzt, die sich nicht aus dem Fließbild ableiten lassen. Danach wird die Simulation in Aspen durchgeführt (3a). Parallel dazu wird bereits das Fließbild weiterbearbeitet und um Trennung und Rückführung ergänzt (3b).
4. Mit Hilfe des Integrationswerkzeugs wird anschließend ein Abgleich zwischen Fließbild und Simulationsmodell durchgeführt. Dabei werden Änderungen bidirektional propagiert. Zum Einen werden die Simulationsergebnisse ins Fließbild zurückzupropagieren. Das Fließbild fungiert also als zentrales Dokument zur Beschreibung der verfahrenstechnischen Anlage, in dem insbesondere auch Simulationsdaten gesammelt werden. Zum Anderen werden die Erweiterungen des Fließbildes in das Simulationsmodell propagiert, so dass im nächsten Schritt eine Gesamtsimulation der Anlage durchgeführt werden könnte. Dabei bleiben die Änderungen am Simulationsmodell erhalten, die der Benutzer manuell durchgeführt hat (Eingabe der Simulationsparameter).

3 CLiP

Leistungsfähige Integrationswerkzeuge setzen voraus, dass formale Modelle zur Beschreibung der im verfahrenstechnischen Entwicklungsprozess erstellten Dokumente existieren. Sonst ließe sich nämlich das erforderliche Wissen über die Struktur und die Inhalte der Dokumente und die zwischen ihnen bestehenden Beziehungen nicht in einem Integrationswerkzeug verankern. Das Integrationswerkzeug könnte dann nur primitive Basisoperationen anbieten, z.B. Erzeugen und Verfolgen von Links, deren Semantik dem Werkzeug unbekannt ist.



a) systemtheoretische Grundlagen

b) Partialmodelle

Abbildung 2: CLiP

Die konzeptionelle Grundlage für die Entwicklung von Integrationswerkzeugen liefert das im SFB 476 entstandene Modellrahmenwerk CLiP (Conceptual Lifecycle Process Model) [1]. CLiP stellt einen wichtigen Meilenstein zur Definition eines integrierten Prozess- und Produktmodells für die Verfahrenstechnik dar - eines der zentralen Ziele des SFB 476. CLiP berücksichtigt bestehende Standards wie PDXI oder PISTEP, verfolgt aber im Gegensatz zu existierenden Standards eher einen Top-Down-Ansatz zur Modellierung: Der Beitrag von CLiP liegt weniger in der detaillierten Ausgestaltung von spezifischen Modellen als in einem neuartigen Rahmenwerk zur Ordnung dieser Modelle.

CliP unterscheidet zwischen verschiedenen Abstraktionsebenen, die durch Instanziierungsbeziehungen miteinander verknüpft werden. Ein Modellelement auf Ebene i dient als Typ für Modellelemente auf der Ebene $i + 1$. Abbildung 2a zeigt die oberste Ebene 0, in der die systemtheoretischen Grundlagen definiert werden, auf denen das gesamte Modellrahmenwerk basiert. Demnach wird ein System durch verschiedene Aspekte beschrieben (Anforderungen, Funktion, Verhalten und Realisierung) und kann Beziehungen zu anderen Systemen haben.

Auf der Ebene 1 werden Modellierungskonzepte eingeführt, die sich in unterschiedlichen Domänen wiederverwenden lassen. So lassen sich beispielsweise Gemeinsamkeiten in der Modellierung von verschiedenen Arten von Fließbildern und Simulationsmodellen identifizieren.

Auf der Ebene 2 werden konkrete Partialmodelle definiert, z.B. für Prozesse, Kostenschätzungen, mathematische Modelle etc. (Abbildung 2b). Die Partialmodelle sind den auf Ebene 0 eingeführten Aspekten zugeordnet (z.B. `Process` dem Funktions- und `Cost` dem Verhaltensaspekt). Auch die in diesem Aufsatz interessierenden Modelle für Fließbilder und Simulationsmodelle werden hier fixiert. Darüber hinaus werden Abhängigkeiten zwischen den Partialmodellen festgelegt, die als Ausgangspunkt für die Definition von Integrationsregeln dienen.

4 Modellbasierte Integration

4.1 Dokumentenmodell

Zum Zugriff auf die zu integrierenden Dokumente und zur Definition von Regeln wird ein zu dem auf Ebene 1 von CLiP definierten Metamodell kompatibles Dokumentenmodell genutzt.

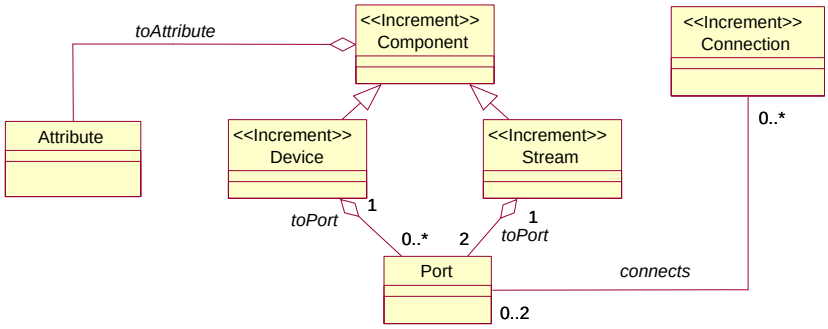


Abbildung 3: Dokumentenmodell als UML-Klassendiagramm

Abbildung 3 zeigt dieses als UML-Klassendiagramm. Die Dokumente enthalten Komponenten (`Component`), die im Falle von fließbildbasierten Dokumenten jeweils entweder Geräte (`Device`) oder Ströme (`Stream`) sind. Geräte haben beliebig viele, Ströme genau zwei Anschlüsse, hier mit `Port` bezeichnet. Diese Anschlüsse können über Verbinder (`Connection`) beliebig miteinander verbunden werden. Komponenten werden über beliebig viele Attribute genauer beschrieben.

Die Datenmodelle der hier zu integrierenden Anwendungen Comos PT und Aspen Plus können leicht auf das hier vorgestellte Metamodell abgebildet werden. Beide Anwendungen verfügen über eigene Strom- und Gerätetypen, die als Instanzen der entsprechenden Metamodell-Klassen betrachtet werden können. In beiden Applikationen wird die Verschaltung der Komponenten wie im Metamodell in Form von Verbindungen zwischen definierten Anschlusspunkten hergestellt.

Die mit dem Stereotyp `Increment` gekennzeichneten Klassen des Metamodells bezeichnen Klassen, deren Instanzen von einem Link im Integrationsdokument referenziert werden können. Objekte der übrigen Klassen werden nicht direkt referenziert, da sie über Aggregationen Teile von anderen Objekten sind.

4.2 Regeldefinition

In CLiP werden bereits Abhängikeiten zwischen den Partialmodellen definiert. Hierbei handelt es sich um einfache 1:1-Beziehungen auf der Typebene, die angeben, welche Elemente des einen Partialmodells welchen Elementen des anderen Partialmodells entsprechen. Eine solche Beziehung kann zum Beispiel angeben, dass der Prozess-Schritt `EnthalpyChangeUO` aus dem Partialmodell `Process` dem Modell `Heater` aus dem Partialmodell `Model Implementation` entspricht. Aus dieser Information lassen sich leicht Regeln ableiten, die von einem Werkzeug zur Integration von Dokumenten genutzt werden können. Umgangssprachlich formuliert wäre das für diesen Fall unter anderem die folgende:

> Wenn im Fließbild eine Wärmetauscher-Komponente vom Typ `EnthalpyChangeUO` vorhanden ist, erzeuge im Simulationsmodell eine `Heater`-Komponente und trage zwischen diesen beiden einen Link ins Integrationsdokument ein.

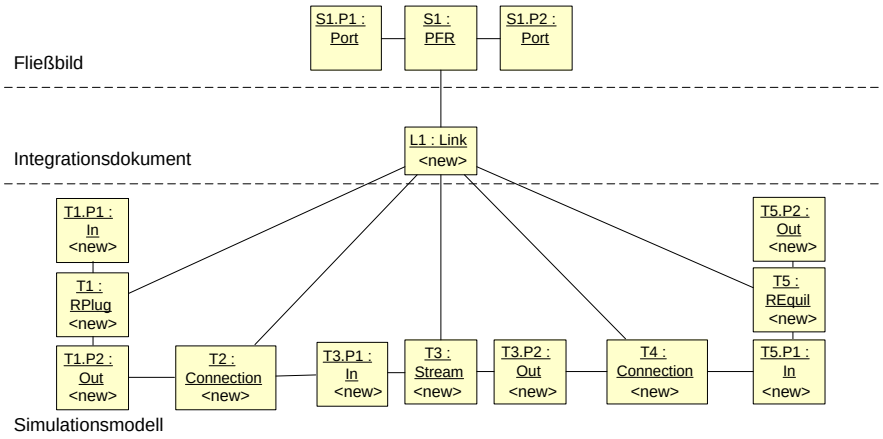


Abbildung 4: Integrationsregel als UML-Objektdiagramm

Diese Regel propagiert eine neue Komponente aus dem Fließbild in die Simulation. Aus der angegeben Beziehung lassen sich auch noch eine Regel ableiten, die Komponenten von der Simulation ins Fließbild überträgt, und eine, die bei in beiden Dokumenten vorhandenen Komponenten deren Übereinstimmung feststellt und nur den zugehörigen Link ins Integrationsdokument übernimmt.

In komplexeren Fällen reichen die Beziehungen auf der Typebene nicht aus, um Integrationsregeln zu definieren. Im Allgemeinen muss daher die Beziehung zunächst auf der Instanzebene in Form von korrespondierenden Mustern genauer spezifiziert werden. Dies ist beispielsweise bei der Reaktion im in Abschnitt 2 beschriebenen Anwendungsszenario der Fall. Hier korrespondiert der Reaktor im Fließbild mit zwei durch einen Strom verbundenen Reaktoren im Simulationsmodell. Dieser Sachverhalt, der mit einfachen Beziehungen auf der Typebene nicht beschreibbar ist, lässt sich durch korrespondierende Muster in Form eines UML-Objektdiagramms beschreiben, wie in Abbildung 4 gezeigt. Im oberen Drittel der Abbildung ist das Muster im Fließbild dargestellt, man erkennt den Reaktor mit seinen beiden Ports. Im unteren Drittel ist das korrespondierende komplexe Muster im Simulationsmodell abgebildet. Die beiden Reaktoren sind über die entsprechenden Ports und einen Strom durch `Connections` verbunden. Die Korrespondenz zwischen den beiden Mustern wird durch ein Objekt der Klasse `Link` dargestellt.

Die spezifizierte Korrespondenz ist aber noch nicht ausreichend, um von einem Integrationswerkzeug genutzt zu werden. Sie gibt lediglich an, welche Muster aus beiden Dokumenten einander entsprechen können. Um eine ausführbare Regel wie im o.a. Beispiel zu erhalten, muss die Korrespondenz noch operationalisiert werden. Daher sind mit dem UML-Stereotyp `new` die Teile des Musters markiert, die vom Integrationswerkzeug erzeugt werden sollen, wenn die unmarkierten Musteranteile in den Dokumenten gefunden worden sind. In diesem Fall bedeutet das, dass, wenn der Reaktor im Fließbild gefunden wird, im Simulationsmodell die beiden über einen Strom verbundenen Reaktoren angelegt werden und im Integrationsdokument ein neuer Link erzeugt wird, der die korrespondierenden Elemente verbindet.

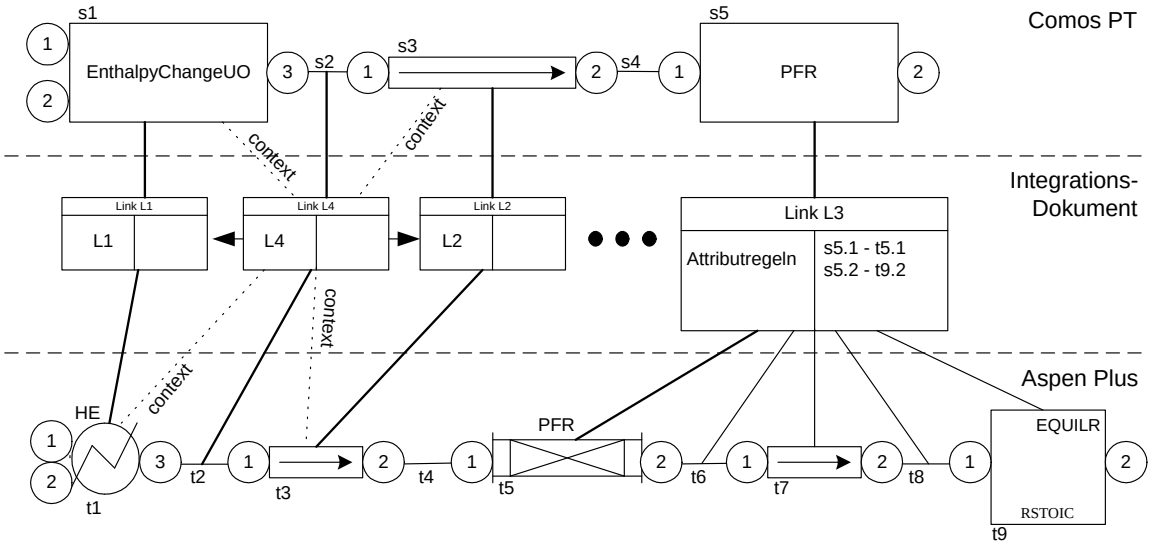


Abbildung 5: Integrationsdokument zum Szenario in Abbildung 1

Die bisherigen Ausführungen bezogen sich auf den Abgleich der Struktur der zu integrierenden Dokumente. Einen wichtigen Aspekt bei der Integration von Fließbild und Simulationsmodell stellt jedoch die Konsistenthaltung der Attribute dar. Jedes Element in beiden Dokumenten ist durch eine Vielzahl von Attributen genauer spezifiziert.

Um den Abgleich von Attributen zu unterstützen, können für jede Regel neben deren strukturellem Anteil Attributregeln definiert werden, die eine Kontrolle der Attribute auf Konsistenz und ggf. eine Zuweisung der Attribute duchführen können. Dabei kann in den Attributregeln auf alle Attribute der Elemente zugegeriffen werden, die nach Anwendung des strukturellen Anteils der Regel durch den entstandenen Link referenziert werden. Die Attributregeln können in Gruppen zusammengefasst werden, von denen eine beim Start des Integrationswerkzeuges ausgewählt wird. Nur die Attributregeln dieser Gruppe werden dann ausgeführt. Somit ist es möglich, für verschiedene Situationen im Entwicklungsprozess entsprechende Regelsätze zu definieren. Beispielsweise können ein Attributregelsatz für die initiale Übertragung des Fließbildes in die Simulation, einer zum Abgleich bei parallel durchgeführten Änderungen und einer zur Rückübertragung von Simulationsergebnissen ins Fließbild definiert werden. Die Attributregeln könnten in UML als OCL-Constraints beschrieben werden, sind aber derzeit anders realisiert (Abschnitt 5).

4.3 Integrationsdokument

Die Abhängigkeiten zwischen den zu integrierenden Dokumenten werden als feingranulare Beziehungen (Link) in einem zusätzlichen Dokument, dem Integrationsdokument abgespeichert. Abbildung 5 zeigt das Integrationsdokument nach der initialen Übertragung des Fließbildes in die Simulation. Im oberen Drittel ist das Fließbild dargestellt, bestehend aus einem Wärmetauscher und einem Reaktor, die über einen Strom verbunden sind. Im unteren Drittel ist die Struktur des Simulationsmodells zu erkennen. Das in der Mitte dargestellte Integrationsdoku-

ment enthält die Links zwischen beiden Dokumenten. Man erkennt am Link L1, dass die beiden Wärmetauscher einander entsprechen. Dieser Link wurde von der ersten im Abschnitt 4.2 beschriebenen Regel erzeugt. Die Anwendung der zweiten beschriebenen Regel führte zum Link L3, der den im Fließbild bereits vorhandenen Reaktor in Beziehung zu den neu erzeugten Elementen im Simulationsmodell setzt. Weitere Links bilden die restlichen Komponenten und die Verbindungen zwischen deren Ports ab.

4.4 Arbeitsweise

Wird das Integrationswerkzeug gestartet, prüft es zunächst die im Integrationsdokument enhaltenen Links auf deren strukturelle Konsistenz. Ein Link ist strukturell konsistent, wenn alle von ihm referenzierten Elemente in den Dokumenten enthalten sind. Inkonsistente Links können gelöscht oder je nach Situation automatisch oder durch den Benutzer wieder in einen konsistenten Zustand überführt werden. Im Anschluss wird die Attributkonsistenz überprüft und ggf. eine Zuweisung der Attribute durchgeführt. Bei der initialen Integration zweier Dokumente ist das Integrationsdokument leer, somit entfällt dieser Schritt.

Danach werden die Dokumente nach Elementen durchsucht, die noch nicht von einem Link referenziert werden. Für diese Elemente wird nun geprüft, ob sie jeweils zu einer der definierten Regeln passen. Ist dies für genau eine Regel der Fall, wird sowohl deren struktureller Teil als auch deren Attributzuweisung angewendet. Sind alternativ für ein oder mehrere Elemente verschiedene Regeln anwendbar, kann der Benutzer zwischen den Alternativen wählen. Neben der Anwendung von Regeln ist es auch möglich, manuell Abhängigkeiten im Integrationsdokument einzutragen. Aus diesen können dann korrespondierende Muster und somit wie in Abschnitt 4.2 beschrieben neue Integrationsregeln abgeleitet werden, die bei späteren Integrationen genutzt werden können (Abschnitt 5).

5 Integrationswerkzeug

Das Integrationswerkzeug zwischen Fließbildern in Comos PT und Simulationsmodellen in Aspen Plus ist als Prototyp implementiert, Abbildung 7 zeigt ihn als PlugIn in Comos PT [3]. Hierzu wurden zunächst die allgemeinen Anteile in einem mehrschichtigen Rahmenwerk zusammengefasst, das zur Implementierung verschiedener Integratoren genutzt werden kann. Abbildung 6 zeigt die grobe Architektur des Sytems. Die beiden zu integrierenden Werkzeuge sind über Wrapper, die von technischen Details der Werkzeuge abstrahieren, an das Integrationswerkzeug angebunden. Dies erleichtert die Anpassung des Integrators an andere Werkzeuge. Der Integratorkern führt die Integration durch, indem er Integrationsregeln anwendet.

Bevor der Integrator in Entwicklungsprozessen eingesetzt werden kann, muss zunächst eine Regelbasis erstellt werden. Hierzu können mit einem auf Rational Rose basierenden UML-Regeleditor nach der in Abschnitt 4.2 vorgestellten Methodik Regeln definiert werden. Diese werden dann vom Integrator interpretiert. Neben der Interpretation der Regeln ist es auch möglich, Regeln in Form von Programmcode fest in den Integrator zu integrieren. Dies bietet sich für Regeln an, die sich nicht ändern und die vom Interpreter nicht unterstützte Funktio-

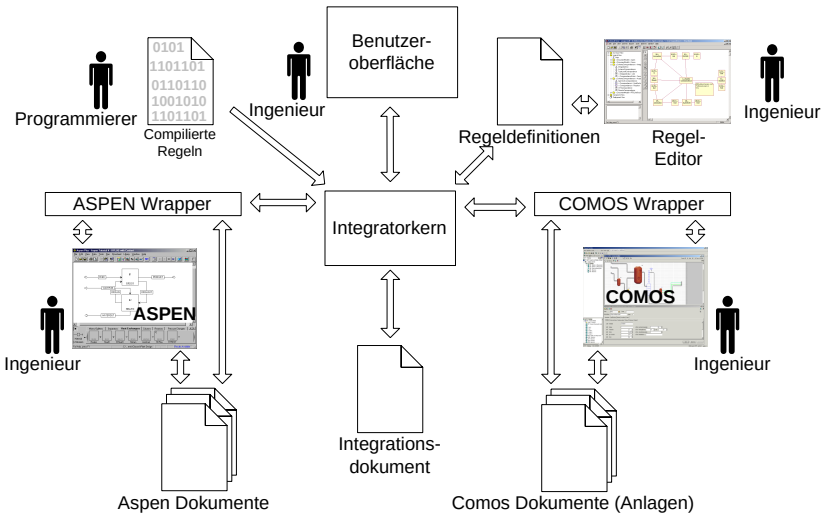


Abbildung 6: Systemarchitektur des Integrationswerkzeugs

nalität benötigen. Der Regelsatz kann später beim Einsatz des Integrators erweitert werden, indem aus bestehenden Links Korrespondenzen und daraus wierderum Regeln abgeleitet werden. Dennoch sollte zumindest für Standardfälle eine Regelbasis vordefiniet werden, um ein unkoordiniertes Wachsen der Regelmenge zu vermeiden. Die Attributregeln werden in der Praxis direkt in Form von Visual-Basic-Skripten definiert, um dem Anwender das Erlernen der OCL zu ersparen. In diesen Skripten kann direkt auf die Elemente der Dokumente zugegriffen werden, wie dies auch bei Skripten innerhalb der beiden Anwendungen der Fall ist.

Im einem verfahrenstechnischen Entwicklungsprozess wird der Integrator immer dann gestartet, wenn die Konsistenz von Fließbild und Simulation überprüft bzw. wiederhergestellt werden soll. Im geschilderten Szenario (Abschnitt 2) ist dies das erste Mal bei der Erstellung des Simulationsmodells der Fall. Während der Integration zeigt der Integrator an der Benutzeroberfläche die anstehenden Benutzerentscheidungen in einer Tabelle an (Abbildung 7 links). Der Benutzer kann entscheiden, in welcher Reihenfolge er die Entscheidungen abarbeitet. Zusätzlich bietet er eine Sicht auf das Integrationsdokument (Abbildung 7 rechts). Hier kann der Benutzer die automatisch erzeugten Links überprüfen und ggf. ändern. Auch das Neuanlegen von Links wird unterstützt. Aus den vom Entwickler bearbeiteten Links können neue Regeln abgeleitet werden. Nach der Integration wird der Integrator beendet und in beiden Werkzeugen wird normal weitergearbeitet. Erst bei einem erneuten Konsistenzabgleich wird der Integrator nochmals gestartet.

6 Fazit

Die Abstimmung zwischen heterogenen Dokumenten in verfahrenstechnischen Entwicklungsprozessen wird von den heute verfügbaren Werkzeugen nur unzureichend unterstützt. Integrationswerkzeuge müssen modell- und regelbasiert arbeiten und das inkrementelle Propagieren von Änderungen erlauben. Voraussetzung dafür ist das explizite Verwalten der Abhängigkeiten zwischen den zu integrierenden Dokumenten.

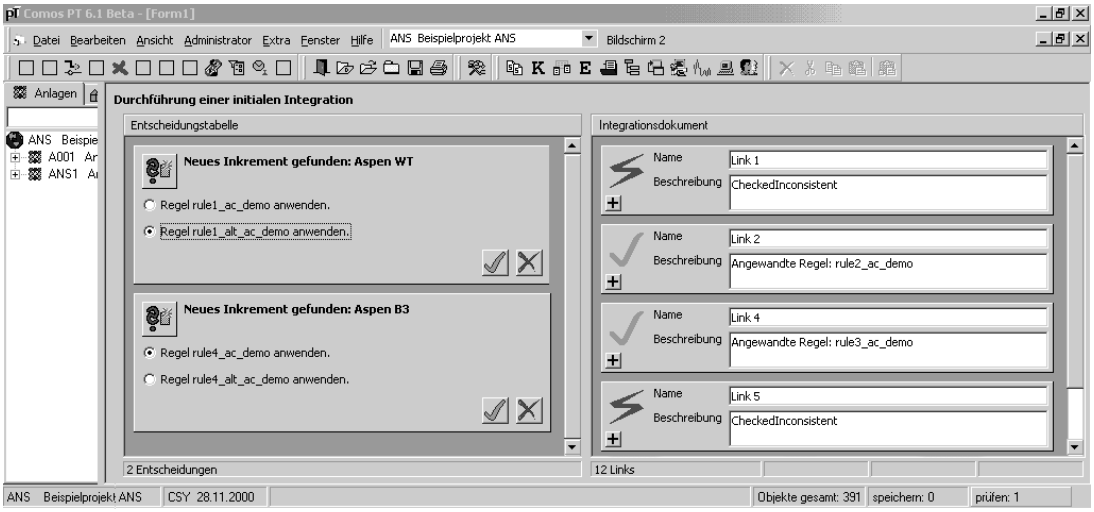


Abbildung 7: Screenshot des Integrators als Comos PT Plug-in

Aktuelle Arbeiten zielen darauf, die Implementierung abzurunden, das implementierte Werkzeug im industriellen Kontext zu validieren und mit Hilfe des Rahmenwerks weitere Integrationswerkzeuge zu erstellen (z.B. für die Integration zwischen Verfahrens- und RI-Fließbildern).

Danksagung

Diese Arbeit wird von der DFG im Rahmen des SFB 476 gefördert. Die Autoren danken Birgit Bayer vom Lehrstuhl für Prozesstechnik der RWTH Aachen für ihre Informationen über CLiP und für ihre Unterstützung beim Ausarbeiten des Anwendungsbeispiels. Wir danken der Innotec GmbH, insbesondere Marcus Elo und Bernd Kokkelink, für die gute Zusammenarbeit.

Literatur

[1] BIRGIT BAYER, RALPH SCHNEIDER und WOLFGANG MARQUARDT: *CLiP - Ein konzeptuelles Rahmenwerk für Produktmodelle*. In: VDI/VDE-Gesellschaft Mess- und Automatisierungstechnik: *Automatisierungstechnik im Spannungsfeld neuer Technologien*, VDI-Berichte 1608, VDI-Verlag, 681-688, 2001.

[2] SIMON BECKER, THOMAS HAASE, BERNHARD WESTFECHTEL und JENS WILHELMS: *Integration Tools Supporting Cooperative Development Processes in Chemical Engineering*. In: *Proceedings of the Sixth Biennial World Conference on Integrated Design and Process Technology (IDPT-2002)*, 10 Seiten, 2002.

[3] SIMON BECKER und JENS WILHELMS: *Integrationswerkzeuge in verfahrenstechnischen Entwicklungsprozessen*. Verfahrenstechnik, 36(6):44–45, Juni 2002.

[4] MANFRED NAGL und BERNHARD WESTFECHTEL (Herausgeber): *Integration von Entwicklungssystemen in Ingenieuranwendungen*. Heidelberg, Germany, 1998.

[5] www.aspentec.com

[6] www.innotec.de