Evaluation von Open-Source-Software für System Dynamics hinsichtlich deren Integrierbarkeit

Axel Hummel[1], Heiko Kern[1], Christian Böhme[2], René Keßler[2] und Arndt Döhler[2]

[1] Betriebliche Informationssysteme, Institut für Informatik, Universität Leipzig
Johannisgasse 26, 04103 Leipzig, Deutschland
{hummel, kern}@informatik.uni-leipzig.de

[2] Intershop Communications AG
Intershop Tower, 07740 Jena, Deutschland
{c.boehme, r.kessler, a.doehler}@intershop.de

Abstract: System Dynamics gehört heutzutage zu den bekanntesten Simulationstechniken und eignet sich insbesondere für die Ableitung langfristig wirksamer Entscheidungsregeln. Im Forschungsprojekt SimProgno wird diese Simulationstechnik für die simulative Prognose von Fragestellungen im E-Commerce eingesetzt. Ziel des Projektes ist die Entwicklung eines Integrations-Frameworks für die Konstruktion komplexer Simulationsmodelle auf Basis von Teilsimulationen, um die in dieser Anwendungsdomäne existierenden multidimensionalen Wirkungszusammenhänge abzubilden. Die Ausführung der Teilsimulationen mithilfe des Frameworks erfordert dazu die Integration spezifischer Laufzeitumgebungen. In diesem Beitrag werden insgesamt zehn Laufzeitumgebungen für System Dynamics auf deren Eignung für SimProgno evaluiert. Besondere Bedeutung kommt hierbei Open-Source-Softwarelösungen zu, deren Quelloffenheit einen großen Vorteil für die Integration darstellt. In der vorliegenden Arbeit wird auf Basis der durchgeführten Evaluation die Open-Source-Software Sphinx SD Tools ausgewählt. Die im Projekt angewendete Methodik für die System Dynamics-Modellentwicklung mit den Sphinx SD Tools und deren Integration in das SimProgno-Framework ist ebenfalls Bestandteil des Beitrags.

1 Einleitung

Computersimulationen sind heutzutage ein wichtiges Hilfsmittel zur Modellierung und Analyse komplexer Systeme. Die Durchführung von Simulationsexperimenten erlaubt die Prognose verschiedener Szenarien ohne das real existierende System zu verändern und trägt damit wesentlich zur Entscheidungsunterstützung bei. Eine der verbreitetsten Simulationstechniken überhaupt ist System Dynamics. Diese Methode erlaubt auf Basis der modellierten Systemstruktur und des dadurch verursachten Verhaltens insbesondere langfristig wirksame Entscheidungsregeln abzuleiten. Bekannt wurde System Dynamics durch das World-Modell, eine für den Club of Rome entwickelte Simulation, in welcher die Wechselwirkungen zwischen den Faktoren Weltbevölkerung, industriellem Wachstum, Umweltverschmutzung, Rohstoffe und Nahrungsmittelproduktion modelliert wurden, um

damit die Auswirkungen auf das Ökosystem der Erde zu prognostizieren [Mea74]. Heute wird System Dynamics in zahlreichen weiteren Anwendungsgebieten eingesetzt. Dazu zählen insbesondere betriebswirtschaftliche Fragestellungen [Ste00], der sozialwissenschaftliche Bereich [GT05], die Finanzwirtschaft [HSHP97] sowie das Gesundheitswesen [Dan99, Bra08].

Im Forschungsprojekt SimProgno [Sim12b] wird diese Simulationstechnik für die simulative Prognose von Fragestellungen im E-Commerce eingesetzt. Die Problemstellungen der Anwendungsdomäne betreffen typischerweise betriebswirtschaftliche, sozialwissenschaftliche sowie technische Aspekte und sind daher durch multidimensionale Wirkungszusammenhänge gekennzeichnet. In SimProgno wird daher der Ansatz verfolgt, die Problemstellungen zunächst in Teilprobleme zu zerlegen und für diese entsprechende Simulationslösungen zu entwickeln. Mithilfe eines integrativen Frameworks werden anschließend komplexe Simulationsmodelle auf Basis der Teilsimulationen konstruiert, um die multidimensionalen Wirkungszusammenhänge abbilden zu können.

Das Integrations-Framework verfügt über eine zentrale Benutzeroberfläche für die Konfiguration der Simulationsmodelle sowie die Auswertung der Simulationsergebnisse. Für die Ausführung der einzelnen Teilsimulationen sind spezifische Laufzeitumgebungen in das Framework zu integrieren. Die Fähigkeit der technischen Integration sowie die dafür erforderliche lizenzrechtliche Erlaubnis stellt somit die wesentliche Voraussetzung für den Einsatz in SimProgno dar. Plattformunabhängigkeit, d.h. die Unterstützung der Betriebssysteme Microsoft Windows, Mac OS und Linux ist eine weitere Anforderung an die Laufzeitumgebungen. Das Integrations-Framework selbst ist in Java implementiert. Idealerweise sollten die Laufzeitumgebungen daher über eine Java-Programmierschnittstelle verfügen. Für System Dynamics existieren heutzutage leistungsfähige Simulationsumgebungen, welche nicht nur über die notwendige Laufzeitumgebung verfügen, sondern zusätzlich eine graphische Benutzeroberfläche für die Erstellung der Simulationsmodelle sowie entsprechende Komponenten für die Auswertung der Simulationsergebnisse beinhalten. Wir haben insgesamt zehn dieser Simulationsumgebungen auf deren Eignung für SimProgno untersucht. Eine besondere Bedeutung kommt dabei Open-Source-Produkten zu, deren Quelloffenheit einen großen Vorteil für die Integration darstellt.

Ziel des Beitrags ist zum einen die Darstellung der Evaluationsergebnisse und zum anderen die Vorstellung der Sphinx SD Tools. Diese Open-Source-Software hat die Anforderungen von SimProgno von allen untersuchten Kandidaten am besten erfüllt. Im nachfolgenden Kapitel werden zunächst die Grundlagen von System Dynamics erläutert. Anschließend erfolgt die Beschreibung der Evaluationsergebnisse. Im vierten Kapitel wird die Open-Source-Simulationsumgebung Sphinx SD Tools vorgestellt. Dabei gehen wir zunächst auf die Modellentwicklung mit den Sphinx SD Tools ein und zeigen im zweiten Teil anhand eines Beispiels, wie diese Software in andere Java-Programme integriert werden kann.

2 Grundlagen von System Dynamics

System Dynamics ist eine Methode für die Analyse komplexer Systeme, welche in den 50er Jahren am MIT durch Jay W. Forrester entwickelt wurde [For71]. Diesem Ansatz liegt die Annahme zugrunde, dass die Struktur eines komplexen Systems dessen Verhalten bestimmt. Das betrachtete System wird daher als Menge von relevanten Systemelementen modelliert, die über Rückkopplungsbeziehungen miteinander verbunden sind. Die Art, wie die einzelnen Systemelemente miteinander in Beziehung stehen, bestimmt das Verhalten des Gesamtsystems über die Zeit [Ste00].

System Dynamics unterstützt sowohl die qualitative als auch quantitative Modellierung komplexer Systeme. Die qualitative Modellierung erfolgt mithilfe von Kausaldiagrammen, welche die Systemkomponenten und deren Beziehungen untereinander in graphischer Form beschreiben. Die Wirkungsbeziehungen werden durch gerichtete Pfeile von der Ursache hin zum Effekt dargestellt. Dabei wird zwischen gleichgerichteten / positiven Wirkungsbeziehungen (mit „+“ bezeichnet) und gegensätzlichen / negativen Wirkungsbeziehungen (mit „–“ bezeichnet) unterschieden. Basierend auf diesen Wirkungsbeziehungen lassen sich komplexe Rückkopplungsschleifen zwischen den Systemelementen erkennen, die entsprechend der Polarität der zugehörigen Wirkungsbeziehungen als positive Rückkopplungsschleifen (dargestellt mittels „R“) und negative Rückkopplungsschleifen (dargestellt mittels „B“) gekennzeichnet werden.

Abbildung 1 zeigt beispielhaft die kausalen Zusammenhänge für die Marktdurchdringung eines neuen Produktes basierend auf dem Bass-Diffusionsmodell [Bas69]. Hierbei wird zwischen Kunden und potenziellen Kunden unterschieden. Die Adoptionsrate gibt den Kundenzuwachs an und wirkt damit positiv auf die Anzahl der Kunden. Die Menge der potenziellen Kunden wird jedoch negativ beeinflusst, da diese Menge mit steigender Adoptionsrate schneller sinkt. Aufgrund der positiven und negativen Wirkungsbeziehung existiert zwischen den potenziellen Kunden und der Adoptionsrate eine als Marktsättigung bezeichnete negative Rückkopplungsschleife. Zwischen den Systemelementen Adoptionsrate, Kunden und Adoptionsrate Mundpropaganda existiert eine zweite Rückkopplungsschleife mit der Bezeichnung Mundpropaganda. Aufgrund der positiven Polarität aller zugehörigen Wirkungsbeziehungen ist diese selbst positiv.

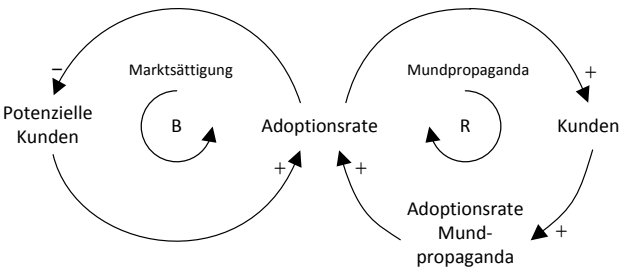

Abbildung 1: Kausaldiagramm für die Marktdurchdringung eines neuen Produktes [Ste01]

Kausaldiagramme stellen eine erste Konzeptualisierung des betrachtenden Systems dar und liefern wichtige Einblicke in die Interaktionsbeziehungen zwischen den einzelnen Systemelementen. Dennoch erlauben Kausaldiagramme keine zahlenmäßigen Aussagen über das Systemverhalten. Für die dafür notwendige quantitative Modellierung werden die Kausaldiagramme in Flussdiagramme überführt indem die Systemelemente und Wirkungsbeziehungen genauer klassifiziert werden [BVS+04].

Flussdiagramme unterscheiden zwischen den vier Systemelementtypen Bestandsgröße, Flussgröße, Quelle oder Senke sowie Hilfsgröße. Die Wirkungsbeziehungen zwischen diesen Systemelementen werden in Materialfluss und Informationsfluss unterteilt. Nach Sterman [Ste00] sind diese Bestandteile wie folgt definiert:

Bestandsgrößen sind Akkumulatoren abzählbarer Entitäten, wie Lagerstände oder Kontostände. Diese werden während des Zeitverlaufs auf- und abgebaut. Eine Bestandsgröße hat zu jedem Zeitpunkt einen Wert, der den Füllstand der Bestandsgröße beschreibt und durch die spezifische Kapazität der Bestandsgröße begrenzt wird.

Flussgrößen beschreiben die Flussraten eines Materialflusses und regulieren damit den Materialstrom zwischen zwei Bestandsgrößen.

Quellen und Senken sind spezielle Bestandsgrößen, welche die Systemgrenzen kennzeichnen. Im Gegensatz zu klassischen Bestandsgrößen besitzen diese keinen Füllstand und verfügen über eine unbegrenzte Kapazität. Materialflüsse werden daher niemals durch Quellen oder Senken eingeschränkt.

Hilfsgrößen sind Systemelemente, die den bisher genannten Elementtypen nicht zugeordnet werden können. Mit Hilfsgrößen werden zum Beispiel systemrelevante Konstanten, Eingabeparameter oder Zwischenergebnisse von Berechnungen modelliert.

Materialflüsse verbinden Bestandsgrößen miteinander und symbolisieren den gerichteten Fluss realer Entitäten zwischen diesen Bestandsgrößen.

Informationsflüsse beschreiben Abhängigkeiten zwischen Systemelementen, die keine Materialflüsse darstellen. Jeder Informationsfluss ist gerichtet und besitzt entweder eine positive oder negative Polarität, analog den Wirkungsbeziehungen der Kausaldiagramme. Zusätzlich können Informationsflüsse Zeitverzögerungen aufweisen.

In der Literatur existieren verschiedene Notationen für Flussdiagramme. Wir verwenden die in Abbildung 2 angegebene graphische Notation von Sterman [Ste00].

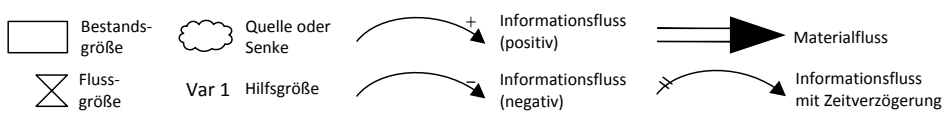

Abbildung 2: Graphische Notation für Flussdiagramme nach [Ste00]

Das Flussdiagramm des Bass-Diffusionsmodells nach [Ste00] ist in Abbildung 3 dargestellt. Die Menge der potenziellen Kunden und die Menge der Kunden werden als Bestandsgrößen modelliert. Der Wechsel von potenziellen Kunden zu tatsächlichen Kunden wird mittels Materialfluss dargestellt. Dieser wird durch eine Adoptionsrate reguliert, die sich aus zwei Teilraten zusammensetzt. Zum einen die Adoption aufgrund externer Faktoren, hier als Adoptionsrate Werbung bezeichnet, zum anderen die Adoption welche durch die Kunden selbst ausgelöst wird, zum Beispiel durch Nachahmungseffekte. Diese als Mundpropaganda bezeichnete Adoption findet statt, wenn potenzielle Kunden auf tatsächliche Kunden treffen und diese von dem neuen Produkt erzählen. Die Menge der sozialen Kontakte von potenziellen Kunden ist dabei abhängig von der Menge der potenziellen Kunden und der allgemeinen Kontaktrate. Beide Hilfsgrößen haben eine positive Wirkung auf die Mundpropaganda. Für die Mundpropaganda ist weiterhin ein Kontakt mit Kunden notwendig. Daher wird die Mundpropaganda von der Kundenanzahl positiv beeinflusst. Von der Gesamtbevölkerung wird die Mundpropaganda jedoch negativ beeinflusst, da durch eine steigende Gesamtbevölkerung die Wahrscheinlichkeit sinkt, einen Kunden zu kontaktieren. Schließlich beschreibt die Adoptionswahrscheinlichkeit die Wirksamkeit der Mundpropaganda, sodass auch dieser Faktor positiv auf die Mundpropaganda wirkt.

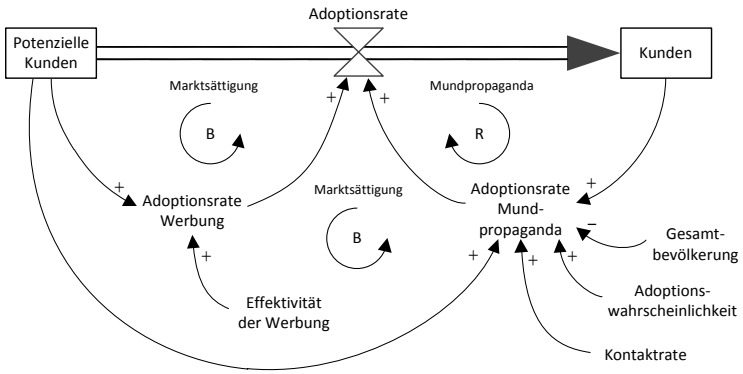

Abbildung 3: Flussdiagramm des Bass-Diffusionsmodells nach [Ste00]

Abschließend wird ein simulationsfähiges Modell auf Basis der Flussdiagramme definiert. Das Verhalten der Bestandsgrößen ist eindeutig durch die Flussdiagramme festgelegt. Der Wert einer Bestandsgröße zum Zeitpunkt t errechnet sich aus dem Wert des vorausgegangenen Zeitpunktes $t-1$, erhöht um die Summe aller materiellen Zuflüsse zum Zeitpunkt t und vermindert um die Summe aller materiellen Abflüsse zum Zeitpunkt t.

Die Bestandsgrößen potenzielle Kunden und Kunden sind daher wie folgt definiert:

$$\frac{d\,(\text{Potenzielle Kunden})}{dt} = -\ \text{Adoptionsrate} \tag{1}$$

$$\frac{d\,(\text{Kunden})}{dt} = \text{Adoptionsrate} \tag{2}$$

Die Durchführung eines Simulationsexperimentes erfordert zudem die Initialisierung der

Bestandsgrößen mit einem Startwert.

Für Flussgrößen und Hilfsgrößen sind mathematische Gleichungen anzugeben, welche das Verhalten dieser Größen beschreiben. Im Bass-Diffusionsmodell sind die Hilfsgrößen Effektivität der Werbung, Gesamtbevölkerung, Adoptionswahrscheinlichkeit und Kontaktrate als Konstanten definiert. Die anderen Systemelemente sind nach [Ste00] wie folgt beschrieben:

$$\text{Adoptionsrate} = \text{Adoptionsrate Werbung} + \text{Adoptionsrate Mundpropaganda} \tag{3}$$

$$\text{Adoptionsrate Werbung} = \text{Potenzielle Kunden} \times \text{Effektivität der Werbung} \tag{4}$$

$$\text{Adoptionsrate Mundpropaganda} = \frac{\text{Potenzielle Kunden} \times \text{Kontaktrate} \times \text{Kunden} \times \text{Adoptionswahrscheinlichkeit}}{\text{Gesamtbevölkerung}} \tag{5}$$

Für die Ausführung des hier beschriebenen Bass-Diffusionsmodells mit den Sphinx SD Tools in Abschnitt 4 verwenden wir die nachfolgend genannten Startwerte und Konstanten.

$$\begin{aligned} \text{Potenzielle Kunden } (t_0) &= 1.000.000 & \text{Kunden } (t_0) &= 0 \\ \text{Effektivität der Werbung} &= 0,011 & \text{Gesamtbevölkerung} &= 1.000.000 \\ \text{Adoptionswahrscheinlichkeit} &= 0,015 & \text{Kontaktrate} &= 10 \end{aligned} \tag{6}$$

3 Simulationssoftware für System Dynamics

Nach Banks et al. [BCINN10] lassen sich drei verschiedene Kategorien von Simulationssoftware unterscheiden. Erstere umfasst die klassischen Programmiersprachen wie C, C++ oder Java. Diese Programmiersprachen wurden nicht für die Implementierung von Simulationsmodellen entwickelt und bieten daher keine besondere Unterstützung für Simulationen an, sodass notwendige Grundfunktionalitäten wie Zeitsteuerung selbst entwickelt werden müssen. Daneben existieren spezialisierte Programmiersprachen für Simulationen wie die General Purpose Simulation System (GPSS) oder AgentSpeak. Diese Programmiersprachen unterstützen spezifische Konzepte einzelner Simulationstechniken, erfordern jedoch das Erlernen der jeweiligen Sprache und erlauben lediglich eine textuelle Modellentwicklung. Die Simulationssoftware mit dem höchsten Abstraktionsniveau sind integrierte Simulationsumgebungen, welche nahezu alle Schritte einer Simulationsstudie unterstützen.

Für System Dynamics existieren heutzutage zahlreiche Simulationsumgebungen. Diese erlauben die graphische Modellierung des zu untersuchenden Systems in Form von Flussdiagrammen ohne Kenntnis einer Programmiersprache. Des Weiteren verfügen diese Simulationsprogramme über eine Laufzeitumgebung für die Definition und Ausführung von Simulationsexperimenten. Die graphische Aufbereitung der Ergebnisse eines Simulationsexperimentes ist ebenfalls auf einfache Art und Weise möglich. Die bekanntesten Simulationsumgebungen für System Dynamics sind die kommerziellen Softwareprogramme Vensim von Ventana Systems [Ven12], Powersim Studio 9 von Powersim Software [Pow12],

Merkmal	Vensim	Powersim	iThink / STELLA	AnyLogic	Consideo Modeler
Version	5.11a	Studio 9	9.1.4 / 9.1.4	6.7.1	7.5.1
Modelle laden	ja	ja	ja	ja	nein
Modelle ausführen	ja	ja	ja	ja	nein
Ergebnisse zurückgeben	ja	ja	ja	ja	nein
Arten der Integration	Batch-Aufruf, API	Batch-Aufruf, API	Batch-Aufruf, API	Batch-Aufruf, API	keine
Technologie / Programmier-sprache	C++, C#	.NET	.NET	Java	(Java)
Betriebssystem	Windows, Mac OS	Windows	Windows, Mac OS	Windows, Linux, Mac OS	Windows, Linux, Mac OS
Ausschluss-kriterium	C++, C#	.NET	.NET	hoher Preis	fehlende Integration

Tabelle 1: Übersicht zu kommerziellen Simulationsumgebungen für System Dynamics und deren Eignung für SimProgno

die Schwesterprodukte iThink [ise12a] und STELLA [ise12b] von isee systems, AnyLogic von XJ Technologies [XJ 12] sowie der Consideo Modeler von der CONSIDEO GmbH [CON12].

Für SimProgno sind diese Simulationsumgebungen jedoch nicht geeignet. Ein Grund dafür ist die fehlende Plattformunabhängigkeit der meisten Softwareprogramme. Vensim, Powersim, iThink und STELLA sind nur für Microsoft Windows Betriebssysteme sowie teilweise für das Betriebssystem Mac OS verfügbar. Die verwendeten Technologien für die Implementierung (C++, C#, .NET) erlauben zudem nur eine eingeschränkte Integration der entwickelten Simulationen in das Java-basierte Integrations-Framework von SimProgno. Ein weiteres Problem sind insbesondere für kleinere Projekte die hohen Kosten der genannten Simulationsumgebungen. Neben dem Anschaffungspreis von teilweise über zehntausend Euro für das Standardprodukt müssen speziell für die Integration der jeweiligen Ausführungsumgebungen in eigene Softwareprodukte weitere Entwicklerprogramme mit teils erheblichen Zusatzkosten erworben werden. Eine Ausnahme bildet der plattformunabhängige Consideo Modeler. Dieser ist jedoch als reines Stand-Alone-Produkt konzipiert, ohne Möglichkeit dieses Programm in andere Softwaresysteme zu integrieren. Somit scheidet auch diese Simulationssoftware für unser Projekt aus.

Eine Übersicht zu den kommerziellen Simulationsumgebungen bzgl. der von uns unter-

suchten Eigenschaften ist in Tabelle 1 dargestellt. Die Zeilen mit den Bezeichnungen *„Modelle laden“*, *„Modelle ausführen“* und *„Ergebnisse zurückgeben“* geben an, ob die jeweilige Funktionalität bei einer Integration der Software zur Verfügung steht. Als mögliche Arten der Integration unterscheiden wir zwischen dem Programmaufruf im Batchbetrieb und dem Zugriff auf die Simulationsmodelle mittels Programmierschnittstelle (API). Die für die API eingesetzte Programmiersprache bzw. Technologie ist in der nachfolgenden Tabellenzeile angegeben. Für eine erfolgreiche Integration mittels API sind diese Technologien daher zwingend zu beherrschen. Die letzte Zeile der Tabelle nennt schließlich den wesentlichen Grund für die Ablehnung der jeweiligen Simulationsumgebung im Kontext von SimProgno.

Open-Source-Simulationsumgebungen für System Dynamics stellen eine vielversprechende Lösungsalternative dar. Diese Softwareprogramme bieten ebenfalls die Möglichkeit einer vollintegrierten Modellentwicklung und sind zudem aufgrund ihrer Quelloffenheit leicht in andere Softwaresysteme zu integrieren. Wir haben die Open-Source-Programme Sphinx SD Tools [Sph12], MapSim [Map12], System Dynamics [Joa12] und Simantics System Dynamics [Sim12a] hinsichtlich unserer Anforderungen in SimProgno untersucht. Alle Softwareprogramme beeinhalten sowohl einen Modelleditor als auch eine integrierte Ausführungsengine. MapSim bietet von den vier Softwareprogrammen den geringsten Entwicklungskomfort. Einerseits erfolgt die Modellkonstruktion mittels textueller Modellierungssprache, andererseits ist keine Auswertung der Simulationsergebnisse in MapSim möglich. Entscheidend für die Ablehnung von MapSim sind jedoch die eingesetzte .NET-Technologie und die fehlende Plattformunabhängigkeit. Die anderen drei Simulationsumgebungen sind alle mittels Java implementiert und sowohl unter Microsoft Windows als auch Linux lauffähig. Die Simulationsumgebung System Dynamics von Simantics setzt auf dem Programm Open Modelica auf und erzeugt daher ausführbaren Modellcode in der Programmiersprache C [LRKY11]. Bei System Dynamics fällt insbesondere die fehlende Programmierschnittstelle (API) sowie der im Vergleich zu den Sphinx SD Tools geringere Reifegrad negativ ins Gewicht. In SimProgno haben wir uns daher für die Verwendung der Sphinx SD Tools entschieden.

Die für unsere Entscheidungsfindung relevanten Eigenschaften der untersuchten Open-Source-Simulationsumgebungen sind in Tabelle 2 angegeben. Auch hier geben die Tabellenzeilen *„Modelle laden“*, *„Modelle ausführen“* und *„Ergebnisse zurückgeben“* an, ob die jeweilige Funktionalität für die integrierte Simulationsumgebung zur Verfügung steht. Die mit *„Technologie / Programmiersprache“* bezeichnete Zeile nennt die für eine Integration notwendigen Programmierkenntnisse. Das Ausschlusskriterium gibt schließlich den Hauptgrund für die Ablehnung des jeweiligen Softwareprogramms an.

Die Evaluation der Open-Source-Simulationsumgebungen wurde auf Basis von funktionalen Eigenschaften durchgeführt. Ein ebenfalls relevantes Evaluationskriterium für Open-Source-Software ist die Lizenz, welcher die Software unterliegt. Auch in dieser Hinsicht sind die Sphinx SD Tools für SimProgno sehr gut geeignet, da die gewählte Apache License Version 2.0 sehr viele Freiheiten lässt. So muss eigene Software, die eine Software verwendet, welche unter der Apache License Version 2.0 steht, nicht unter diese Lizenz gestellt werden. Weiterhin wird nicht gefordert, dass Änderungen am Quellcode von Software, die unter dieser Lizenz steht, dem Lizenzgeber zugänglich gemacht werden [Apa04].

Merkmal	Sphinx SD Tools	MapSim	System Dynamics	Simantics System Dynamics
Version	0.7b	4.1	1.3	1.4
Modelleditor	graphisch	textuell	graphisch	graphisch
Modelle laden	ja	ja	ja	ja
Modelle ausführen	ja	ja	ja	ja
Ergebnisse zurückgeben	ja	ja	ja	ja
Arten der Integration	API	Batch-Aufruf, API	Batch-Aufruf	Batch-Aufruf, API
Technologie / Programmier-sprache	Java	.NET	Java	C
Betriebssystem	Windows, Linux, Mac OS	Windows	Windows, Linux, Mac OS	Windows, Linux, Mac OS
Lizenz	Apache License 2.0	LGPL Version 2.0	GPL Version 2.0	EPL
Ausschluss-kriterium	keines	.NET	Reifegrad, API	C

Tabelle 2: Übersicht zu Open-Source-Simulationsumgebungen für System Dynamics und deren Eignung für SimProgno

4 Die Open-Source-Software Sphinx SD Tools

In diesem Abschnitt beschreiben wir die grundlegenden Funktionalitäten der Sphinx SD Tools. Dabei gehen wir zunächst auf die Modellentwicklung von System Dynamics unter Verwendung der Sphinx SD Tools ein und erläutern anschließend, wie die konstruierten Simulationsmodelle auch in anderen Java-Programmen verwendet werden können.

4.1 Modellentwicklung mit den Sphinx SD Tools

Die Entwicklung von System Dynamics-Modellen ist ein iterativer Prozess, bestehend aus den Phasen qualitative Modellkonstruktion, quantitative Modellkonstruktion, Definition und Ausführung der Simulationsexperimente sowie Auswertung der Simulationsergebnis-

se. Die Modellkonstruktion mit den Sphinx SD Tools erfolgt in Form von Flussdiagrammen mithilfe einer graphischen Benutzeroberfläche. Die Sphinx SD Tools unterscheiden dabei zwischen Bestandsgrößen, Flussgrößen und Hilfsgrößen sowie zwischen Materialflüssen und Informationsflüssen. Die graphischen Symbole weichen von der in Abbildung 2 angegebenen Notation ab. Abbildung 4 zeigt das mit den Sphinx SD Tools konstruierte Flussdiagramm des Bass-Diffusionsmodells. Neben dem graphischen Modelleditor beinhalten die Sphinx SD Tools zusätzlich eine textuelle Repräsentation des System Dynamics-Modells im XML-Format sowie eine Baumdarstellung. Auch über diese Darstellungsformen kann das Simulationsmodell bearbeitet werden.

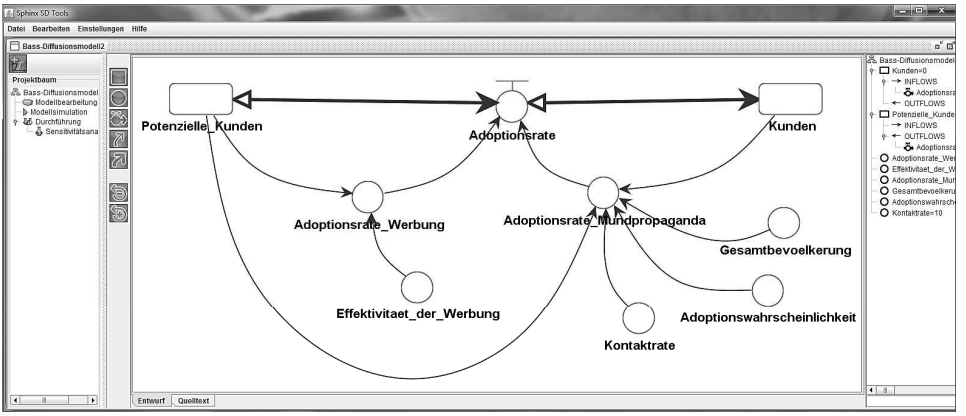

Abbildung 4: Flussdiagramm des Bass-Diffusionsmodells im graphischen Modelleditor der Sphinx SD Tools

Im zweiten Schritt wird das Flussdiagramm zu einem vollständigen mathematischen Modell erweitert. Für diese Aufgabe bieten die Sphinx SD Tools einen leistungsfähigen graphischen Formeleditor, welcher bereits zahlreiche mathematische Funktionen vordefiniert. Zusätzlich können für Systemelemente beliebige Funktionen mittels Wertetabelle definiert werden. Die Wertetabelle wird sowohl als graphischer Funktionsverlauf als auch in tabellarischer Form angezeigt und kann in beiden Ansichten bearbeitet werden. Ein Beispiel dafür ist in Abbildung 5 dargestellt.

Eine Besonderheit der Sphinx SD Tools liegt in der einfachen Erweiterbarkeit des Modelleditors und des Formeleditors. Der Modelleditor erlaubt die Definition von eigenen Templates, inklusive mathematischem Modell und ermöglicht so die Wiederverwendung von Systemstrukturen, wie die von Senge [Sen97] eingeführten Systemarchetypen oder die generischen Systemmoleküle [Hin05]. Der Formeleditor ermöglicht die Spezifikation von benutzerdefinierten Funktionen. Dabei kann auf die vollständige Ausdrucksmächtigkeit von Java zurückgegriffen werden.

Nachdem das Simulationsmodell vollständig spezifiziert ist, erfolgt durch Festlegung der Start- und Endzeit die Definition eines Simulationsexperiments. Die Simulationsergebnisse können wahlweise als Tabelle und / oder in Form von Diagrammen dargestellt werden.

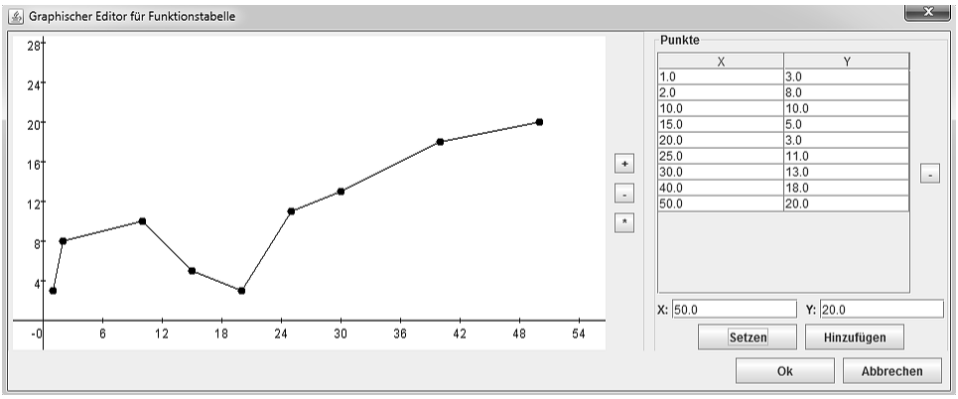

Abbildung 5: Editor für die Spezifikation benutzerdefinierter Funktionen

Die Simulationsergebnisse des Bass-Diffusionsmodells basierend auf den mathematischen Gleichungen (1) bis (6) für eine Simulationsdauer von 50 Zeiteinheiten sind in Abbildung 6 exemplarisch für die beiden Systemgrößen *Potenzielle_Kunden* und *Kunden* dargestellt. Wie erwartet steigt die Anzahl der Kunden immer weiter an, bis schließlich alle potenziellen Kunden das neu eingeführte Produkt angenommen haben. Die Anzahl der potenziellen Kunden geht dementsprechend immer weiter zurück. Sowohl die tabellarischen Ergebnisse als auch die Diagramme können aus den Sphinx SD Tools exportiert werden, um diese weiterzuverarbeiten.

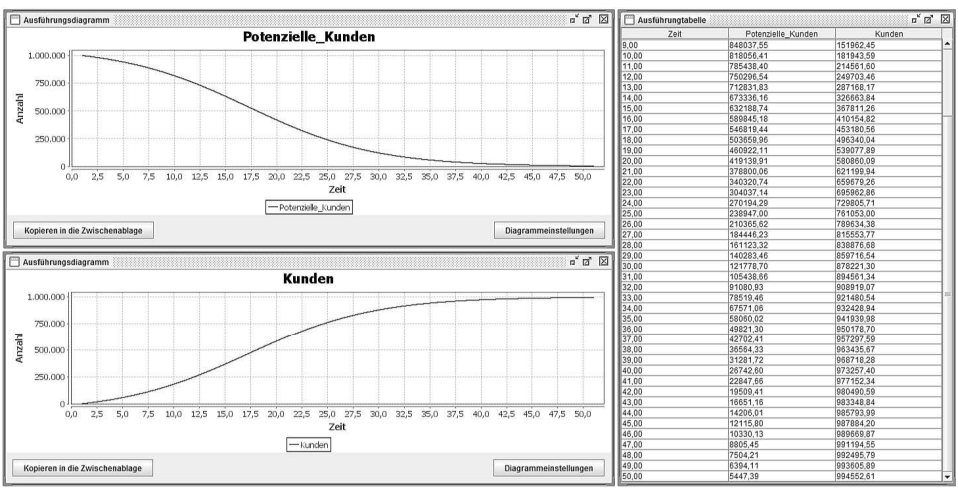

Abbildung 6: Graphische und textuelle Darstellung der Simulationsergebnisse für die beiden Systemgrößen *Potenzielle_Kunden* und *Kunden*

4.2 Modellkonfiguration und -ausführung mithilfe der Sphinx SD Tools API

Zusätzlich zu der eigentlichen Simulationsausführung in den Sphinx SD Tools, kann das Simulationsmodell auf zwei weitere Arten verwendet werden. Zum einen ermöglichen die Sphinx SD Tools einen Modellexport als ausführbares Java-Programm in Form einer JAR-Datei, zum anderen kann das Simulationsmodell mithilfe der bereitgestellten API in andere Softwaresysteme integriert werden.

Für unseren Anwendungsfall sind insbesondere die integrierte Modellmanipulation, deren Ausführung und die Rückgabe der Simulationsergebnisse mittels API von Bedeutung, um die Konfiguration der System Dynamics-Modelle sowie die Auswertung der Simulationsergebnisse mithilfe des SimProgno-Framework durchführen zu können. Nachfolgend wird die Arbeit mit der Sphinx-API an einem Beispiel erläutert. Dabei soll zunächst der Wert für die Systemgröße *Effektivitaet_der_Werbung* im Bass-Diffusionsmodell auf einen benutzerdefinierten Wert gesetzt werden, welcher beispielsweise durch Ausführung einer anderen Simulation ermittelt werden könnte. Anschließend wird ein neues Simulationsexperiment erzeugt und ausgeführt sowie die prognostizierte Kundenentwicklung abgerufen. Listing 1 zeigt den Quellcode des Beispiels.

Zunächst wird das in einer SD-Projekt-Datei (.sdp) gespeicherte Simulationsmodell unter Angabe des Speicherpfades (hier: *C:/sphinxesSD/models*) und der entsprechenden Datei (hier: *BassDiffusionsmodell.sdp*) geladen. Dies erfolgt mittels `ProjectManager`, welcher unter Verwendung der Methode `getProject()` das zugehörige Projekt zurückliefert. Ein Sphinx-Projekt besteht dabei aus vier Bestandteilen. Für uns sind insbesondere die beiden Elemente `ModellingSettings` und `SDModel` von Bedeutung. Das Objekt `ModellingSettings` umfasst die Einstellungen für die Ausführung eines Simulationsexperimentes wie Startzeit, Endzeit und Schrittweite. In Listing 1 wird die Startzeit mit 1, die Endzeit mit 50 und eine Schrittweite von 0,01 festgelegt. Das Objekt `SDModel` beinhaltet die eigentlichen Modellelemente und ermöglicht zum Beispiel die Manipulation von Berechnungsvorschriften. Die Konstante *Effektivitaet_der_Werbung* wird im Beispiel mithilfe der Methoden `getEntityByName()` und `getFormula()` auf den Wert 0,022 gesetzt.

Für die eigentliche Ausführung des Simulationsexperimentes wird ein `ModelExecutor` benötigt. Dieser ist unter Angabe der Objekte `ModellingSettings` und `SDModel` zu erzeugen. Der `ModelExecutor` kann nun mit den beiden Methoden `start()` und `init()` initialisiert werden. In diesem Schritt werden u.a. alle mathematischen Berechnungsvorschriften des Simulationsmodells auf syntaktische Korrektheit geprüft. Mithilfe der Methode `doTick()` kann das Experiment nun schrittweise mit der angegebenen Schrittweite ausgeführt werden, bis die Endzeit erreicht ist. Die Werte der Modellelemente können dabei entweder nach jedem Tick oder nach Ausführung des vollständigen Simulationsexperimentes mittels einer `Map` für den gesamten Simulationszeitraum abgefragt werden. In Listing 1 wird die zweite Variante für die Modellgröße *Kunden* durchgeführt. Die erhaltenen Werte können anschließend weiter verarbeitet werden und als Eingangsgrößen für andere Simulationen dienen.

Listing 1: Anwendung der Java-API der Sphinx SD Tools zur Konfiguration und Ausführung des Bass-Diffusionsmodells

```
1  String path = "C:/sphinxesSD/models";
2  String fileName = "BassDiffusionsmodell";
3  String userDefinedValue = "0.022";
4  // load project
5  ProjectManager projectManager = ProjectManager.newProjectManager(path,
       fileName);
6  Project project = projectManager.getProject();
7  // manipulate model properties
8  SDModel model = project.getModel();
9  model.getEntityByName("Effektivitaet_der_Werbung").getFormula().setText(
       userDefinedValue);
10 // set execution settings
11 ModellingSettings executionSettings = project.getExecutionSettings();
12 executionSettings.setStartTime(1.0);
13 executionSettings.setEndTime(50.0);
14 executionSettings.setDT(0.01);
15 // initialize model
16 ModelExecutor executor = new ModelExecutor(model, executionSettings);
17 executor.start();
18 executor.init();
19 // execute model
20 for(double d = executionSettings.getStartTime(); d <= executionSettings.
       getEndTime(); d+= executionSettings.getDT()) {
21         executor.doTick();
22 }
23 // get simulation results
24 Map<Double, Double> timeline = executor.getEntityExecutor("Kunden").
       getTimeline();
25 // finalize model execution
26 executor.stop();
```

5 Zusammenfassung und Ausblick

Die Konstruktion, Simulation und Auswertung von System Dynamics-Modellen erfolgt mithilfe leistungsfähiger Simulationsumgebungen, deren bekannteste Vertreter allesamt kommerzielle Softwareprodukte sind, die teilweise seit über 20 Jahren auf dem Markt existieren. Dennoch sind heutzutage einige Open-Source-Produkte verfügbar, die aufgrund ihrer Quelloffenheit für die Integration in andere Softwaresysteme besonders geeignet sind. In diesem Beitrag wurden vier Open-Source-Simulationsumgebungen auf deren Integrationsfähigkeit, Implementierungstechnologien und Plattformunabhängigkeit untersucht. Die erzielten Ergebnisse zeigen, dass diese Open-Source-Produkte eine echte Alternative zu kommerziellen Simulationsumgebungen darstellen. Basierend auf der Evaluation haben wir die Open-Source-Software Sphinx SD Tools für den Einsatz im SimProgno-Projekt ausgewählt. Die Methodik zur Entwicklung der Simulationsmodelle sowie deren Integration in andere Softwaresysteme unter Verwendung der Sphinx SD Tools werden in diesem Beitrag beschrieben.

In SimProgno konnten bereits erste Teilsimulationen mit den Sphinx SD Tools realisiert werden. Im weiteren Projektverlauf soll das in Abschnitt 4 beschriebene Vorgehen angewendet werden, um weitere System Dynamics-Simulationen im E-Commerce-Bereich zu entwickeln und mithilfe der API an das Integrations-Framework anzubinden. Dabei soll insbesondere der Template-Mechanismus der Sphinx SD Tools genutzt werden, um

E-Commerce-spezifische Systemstrukturen für die Modellierung zu verwenden. Des Weiteren ist geplant, die Quelloffenheit der Sphinx SD Tools zu nutzen, um die API zu erweitern. Derzeit ist es nicht möglich, die Werte einzelner Modellelemente per API-Zugriff zu verändern, nachdem die Simulation initialisiert wurde. Für die Realisierung komplexer Simulationsszenarien in denen sich einzelne Teilsimulationen wechselseitig zur Laufzeit beeinflussen ist dies jedoch notwendig.

Abschließend bleibt zu wünschen, dass die jungen Open-Source-Initiativen konsequent weiterentwickelt werden und ihre Community stetig vergrößern können. Wir hoffen, dass diese Veröffentlichung einen Beitrag dazu leisten kann.

Danksagung

Dieser Beitrag entstand im Rahmen des Verbundprojekts SimProgno (Förderkennzeichen: 01IS10042C). SimProgno wird gefördert vom Bundesministerium für Bildung und Forschung.

Literatur

[Apa04] Apache Software Foundation. Apache License, Version 2.0, Januar 2004. http://www.apache.org/licenses/LICENSE-2.0.html. Zuletzt geprüft am 20.06.2012.

[Bas69] Frank M. Bass. A New Product Growth for Model Consumer Durables. *Management Science*, 15(5):215–227, 1969.

[BCINN10] Jerry Banks, John S. Carson II, Barry L. Nelson und David M. Nicol. *Discrete-Event System Simulation*. Prentice Hall, Upper Saddle River, N.J. and Singapore, 5. Auflage, 2010.

[Bra08] Sally C. Brailsford. System dynamics: What's in it for healthcare simulation modelers. In Scott J. Mason, Raymond R. Hill, Lars Mönch, Oliver Rose, Thomas Jefferson und John W. Fowler, Hrsg., *Proceedings of the 2008 Winter Simulation Conference, WSC 2008, Miami, Florida, USA, December 7-10, 2008*, Seiten 1478–1483, 2008.

[BVS+04] Thomas Binder, Andreas Vox, Belyazid Salim, Hördur Haraldsson und Mats Svensson. Developing System Dynamics Models from Causal Loop Diagrams. In Michael Kennedy, Graham W. Winch, Robin S. Langer, Jennifer I. Rowe und Joan M. Yanni, Hrsg., *Proceedings of the 22nd International Conference of the System Dynamics Society, Oxford, UK, July 25-29, 2004*, 2004.

[CON12] CONSIDEO GmbH. Consideo Modeler Version 7.5.1, 2012. http://www.consideo-modeler.de/. Zuletzt geprüft am 26.03.2012.

[Dan99] B. C. Dangerfield. System Dynamics Applications to European Health Care Issues. *The Journal of the Operational Research Society*, 50(4):345–353, 1999.

[For71] Jay W. Forrester. Counterintuitive Behavior of Social Systems. *Theory and Decision*, 2(2):109–140, 1971.

[GT05] G. Nigel Gilbert und Klaus G. Troitzsch. *Simulation for the Social Scientist*. Open University Press, Maidenhead, England and New York, NY, 2. Auflage, 2005.

[Hin05] Jim Hines. Molecules of Structure: Building Blocks for System Dynamics Models Version 2.02, 2005. Online verfügbar unter: http://www.systemswiki.org/images/a/a8/Molecule.pdf.

[HSHP97] Christiaan Heij, Hans Schumacher, Bernard Hanzon und Kees Praagman, Hrsg. *System dynamics in economic and financial models*. J. Wiley & sons, 1997.

[ise12a] isee systems, inc. iThink Version 9.1.4, 2012. http://www.iseesystems.com/. Zuletzt geprüft am 26.03.2012.

[ise12b] isee systems, inc. STELLA Version 9.1.4, 2012. http://www.iseesystems.com/. Zuletzt geprüft am 26.03.2012.

[Joa12] Joachim Melcher. System Dynamics Version 1.3, 2012. http://system-dynamics.sourceforge.net/. Zuletzt geprüft am 26.03.2012.

[LRKY11] Teemu Lempinen, Sampsa Ruutu, Tommi Karhela und Peter Ylén. Open Source System Dynamics with Simantics and OpenModelica. In James M. Lyneis und George P. Richardson, Hrsg., *Proceedings of the 29th International Conference of the System Dynamics Society, Washington, DC, USA, July 24-28, 2011*, 2011.

[Map12] MapSim Project. MapSim Version 4.1, 2012. http://mapsim.sourceforge.net/. Zuletzt geprüft am 06.04.2012.

[Mea74] Dennis L. Meadows. *Dynamics of Growth in a Finite World*. Wright-Allen Press, Cambridge, Mass, 1974.

[Pow12] Powersim Software AS. Powersim Studio 9, 2012. http://www.powersim.com/. Zuletzt geprüft am 26.03.2012.

[Sen97] Peter M. Senge. *Die fünfte Disziplin: Kunst und Praxis der lernenden Organisation*. Klett-Cotta, Stuttgart, 4. Auflage, 1997.

[Sim12a] Simantics. Simantics System Dynamics Version 1.4, 2012. https://www.simantics.org/simantics. Zuletzt geprüft am 26.03.2012.

[Sim12b] SimProgno – Ein integratives Framework zur multidimensionalen Modellierung, Simulation und Prognostik von E-Commerce-Wirkungsketten, 2012. http://www.simprogno.de/. Zuletzt geprüft am 26.03.2012.

[Sph12] Sphinxes.org. Sphinx SD Tools Version 0.7b, 2012. http://www.sphinxes.org/. Zuletzt geprüft am 26.03.2012.

[Ste00] John D. Sterman. *Business Dynamics: Systems Thinking and Modeling for a Complex World*. Irwin/McGraw-Hill, Boston, 2000.

[Ste01] John D. Sterman. System Dynamics Modeling: Tools for Learning in a Complex World. *California Management Review*, 43(4):8–25, 2001.

[Ven12] Ventana Systems, Inc. Vensim Version 5.11a, 2012. http://www.vensim.com/. Zuletzt geprüft am 26.03.2012.

[XJ 12] XJ Technologies Company. AnyLogic Version 6.7.1, 2012. http://www.xjtek.com/. Zuletzt geprüft am 26.03.2012.