Proposal zur Diplomarbeit

„Generierung von graphischen Strukturediteren aus visuellen Spezifikationen“

Bastian Cramer – bcramer@upb.de

Einleitung

Visuelle Sprachen spielen eine bedeutende Rolle bei der Modellierung von Softwaresystemen und beim Einsatz in speziellen Anwendungen (DSL = Domain Specific Language) [7]. Durch den Einsatz von bestimmten graphischen Repräsentationen und visuellen Mustern können komplexe Strukturen, wie beispielsweise Algorithmen oder größere Fallunterscheidungen wie in Nassi-Shneiderman Diagrammen so dargestellt werden, dass sie von Menschen leicht erkannt und verstanden werden können. Hierzu werden spezielle graphische Editoren benutzt.

DEViL (Development Environment for Visual Languages) [1] generiert solche. DEViL wird in der FG Programmiersprachen und Übersetzer von Prof. Kastens entwickelt und ist eine Entwicklungsumgebung für visuelle Sprachen.

In der Diplomarbeit soll eine visuelle Spezifikationssprache für DEViL entwickelt werden, die es dem Benutzer ermöglicht, sowohl die abstrakte Struktur, als auch die graphischen Muster, die DEViL benutzt, visuell anzuwenden. Der Benutzer soll bei der Modellierung unterstützt werden um Fehler zu vermeiden.

Vorteil dieses visuellen Frontends ist es, dass es auch Anfängern ermöglicht werden soll, mit DEViL visuelle Sprachen zu entwickeln.

Grundlagen

DEViL ist ein an der FG Programmiersprachen und Übersetzer von Prof. Kastens entwickelte Entwicklungsumgebung für visuelle Sprachen. Die visuellen Umgebungen werden aus Spezifikationen der Sprachstruktur und der graphischen Darstellung der visuellen Sprache erzeugt. Zu Beginn wird dazu das Sprachmodell der visuellen Sprache, die sogenannte abstrakte Struktur, spezifiziert. Die Spezifikation des Modells basiert dabei auf dem Konzept von Klassenhierarchien. Es können Klassen - inklusive einer Vererbungshierarchie -, Attribute, Referenzen und Listen von Unterelementen modelliert werden.

Aufbauend auf dem Sprachmodell, das als Basis dient, werden visuelle Sichten und Codegenerierung spezifiziert. Die Sichten legen visuelle Repäsentationen der abstrakten Struktur, also das Aussehen fest. Die visuellen Sichten basieren dabei auf einem Strukturbaum, der aus der abstrakten Struktur berechnet wird. Dieser Strukturbaum besteht aus Knoten, die den Klassen der abstrakten Struktur entsprechen und ihren Attributen. Auch können den Sichten zusätzliche – nur hier spezifizierte - Knoten hinzugefügt werden, die andere Knoten zusammenfassen und spezielle Kontexte bereitstellen um zusätzliche Muster anzuwenden. Dies dient der Neustrukturierung der Sichten.
Es können so zum Beispiel Listen, Formulare oder Tabellen als sogenannte visuelle Muster eingesetzt werden. Es stehen auch Spezialmuster wie Matrizen, Bäume oder Tabellen zur

Verfügung. Mittels dieser Repräsentation können Änderungen an der abstrakten Struktur vorgenommen werden. Ändert sich die abstrakte Struktur wird die Sicht neu berechnet.

Visuelle Muster werden angewendet, indem Grammatiksymbole von bestimmten Symbolrollen erben. Zusätzlich kann man visuellen Mustern bestimmte generische Zeichnungen zuordnen. Visuelle Muster sind somit nicht auf eine statische Repräsentation begrenzt. Ein Listenmuster kann durch verschiedene generische Zeichnungen mehrere graphische Repräsentationen darstellen.

Anforderungen

In der Diplomarbeit sollen Repräsentationen für die abstrakte Struktur und das Anwenden der Sichten gefunden werden. Es soll untersucht werden, ob die visuelle Darstellung aufgrund der abstrakten Struktur automatisch erstellt werden kann, also ob eine Synchronisation zwischen beiden Teilen möglich ist oder ob die Modellierung von abstrakter Struktur und Darstellung unabhängig voneinander erstellt werden muß. Zu einer gegebenen abstrakten Strukturdefinition sollen verschiedene Sichten spezifiziert werden können.
Visuelle Muster und Endstücke sollten dabei automatisch vervollständigt werden, wenn der Benutzer an einer Stelle ein Muster zuweist. Falsch angewendete oder noch fehlende Muster sollten dem Benutzer ebenfalls angezeigt werden.
Fehler- und Hinweismeldungen sollten verständlich sein und schnell zu einer korrekten visuellen Darstellung führen.

In DEViL können zudem textuelle Repräsentationen der Struktur in einer speziellen Sprache erstellt werden. Diese sollte auch visuell modelliert werden können. Ebenso sollten generische Zeichnungen integriert werden können.

Wenn die visuelle Darstellung komplett ist, muß eine Codeerzeugung entworfen werden, die den entsprechenden LIDO Code generiert.

Auch soll es dem Entwickler möglich sein, später definierte Muster in einer Bibliothek hinzuzufügen und Änderungen an bestehenden Mustern in einer Konfigurationsdatei vornehmen zu können ohne dabei die Programmierung des Frontends zu verändern.

Vorgehensweise

Zunächst muß evaluiert werden, wie ein solches System auszusehen hätte.
Hier sollten auch andere Systeme wie MetaEdit+ [3] und GenGed [4] untersucht werden. Es ist im Besonderen zu beachten, wie die visuellen Muster hier am geeignetsten spezifiziert werden können.
Desweiteren sollten diverse Diagrammarten als potentielle Kandidaten herangezogen werden. Hier sind ER Diagramme und UML zu nennen oder eine Erweiterung von UML sowie eventuell eine ganz neue Art der Repräsentation. Vor allem die Klassendiagramme von UML [2] scheinen geeignet zu sein, die abstrakte Struktur zu modellieren, da diese stark an einen klassenbasierten Entwurf erinnert.

Für das Anwenden der visuellen Muster soll im Anschluß ebenfalls eine Darstellung gefunden werden. Klassendiagramme sind hier wohl nicht geeignet, da Attribute nicht als Knoten dargestellt werden und die Darstellung zu unübersichtlich ist.

Zum Schluß der Diplomarbeit soll eine kurze Evaluationsphase stattfinden in der

verschiedene Probanden Aufgaben lösen, sowohl mit als auch ohne visuelles Frontend. Hier soll gezeigt werden, dass das Frontend einen Geschwindigkeits- sowie einen Sicherheitsvorteil gegenüber der herkömmlichen Entwurfsmethodik erbringt. Evaluiert werden soll auch, ob das Erlernen des DEViL Systems Anfängern durch das Frontend erleichtert wird.

Zeitplan:

1.06 - 30.06.05	Einarbeitungsphase, Evaluation bestehende Systeme, Ideensammlung
23.6 – 7.07.05	Schreiben: Einleitung, Problembeschreibung
7.7.- 30.7.05	Sprachentwurf
23.7.- 8.8.05	Schreiben: Sprachentwurf
8.8.	Entwurf der Ausarbeitung der bisherigen Teile
8.8.- 8.10.05	Implementierung
1.10– 20.10	Schreiben
20.10.	Entwurf der fertigen Ausarbeitung für Carsten
31.10	Entwurf der fertigen Ausarbeitung für Prof. Kastens
20.10.-31.10.05	Evaluation
20.10- 31.11.05	Schreiben: Rest
1.12	Abgabe

Literaturverzeichnis

[1] Carsten Schmidt und Uwe Kastens. Implementation of visual languages using pattern-based specifications. Software - Practice and Experience, 33:1471-1505, Dezember 2003.

[2] Jaeckle, M.: UML 2 Glasklar, Hanser Verlag, 2005

[3] MetaEdit+, http://www.metacase.com

[4] Bardohl,R. et al.: GenGED - A visual definition tool for visual modeling environments Proc. Application of Graph Transformations with Industrial Relevance (AGTIVE'03), pages 407-414, Sept./Oct., 2003, Charlottesville/Virgina, USA.

[5] Schiffer, S.: Visuelle Programmierung, 1998, Addison-Wesley

[6] PG-Wavis: Projektgruppe PaderWAVE, Generierung von Webanwendungen aus visuellen Spezifikationen: Abschlußbericht, 2005

[7] Stahl, T. et al.: Modellgetriebene Softwareentwicklung, Dpunkt Verlag, 2005