

# Typed Server Pages

Am Fachbereich Mathematik und Informatik  
der Freien Universität Berlin  
eingereichte Dissertation  
zur Erlangung des akademischen Grades  
eines Doktors der Naturwissenschaften  
vorgelegt von Dirk Draheim  
Berlin, 2002  
Tag der Disputation: 29.November 2002

Gutachter: Prof. Dr. Elfriede Fehr  
PD Dr. Martin Große-Rhode



# Abstract

In this dissertation a statically typed server pages approach is elaborated. The resulting notions of correctness ensure the type-safe interplay of dynamically generated web forms and targeted software components on the one hand side and the generation of documents that are valid with respect to a given user interface description language on the other hand. The approach contributes the notion of tag support for gathering complex user defined data, the notion of exchanging complex objects virtually across the web user agent, the notion of typed server-side calls to server pages, and the notion of higher order server pages.

Web applications are ubiquitous. Ultra-thin client based tiered enterprise applications become more and more important. Due to this a considerable number of innovative web technologies has been provided recently. Currently web applications and web based system architecture are fields of intensive research activity. Server pages technology is a state-of-the-art in this field. The contributions of this work target stability, maintainability and reusability of server pages based systems. The findings are programming language independent, a concrete programming language can be amalgamated with the found concepts in a way that is conservative with respect to the language's semantics.

In this dissertation a server pages based presentation layer is characterized abstractly as a closed collection of typed dialogue methods. Based on this coding guidelines and rules are defined that together informally provide the desired notions of type correctness and description correctness. The static semantics of the new server pages approach is defined as a Per Martin-Löf style type system with respect to an amalgamation with a minimal imperative programming language and a sufficiently complex equi-recursive type system. The "model two architecture" of web based system applications is analyzed in order to provide a further justification of the proposed approach. NSP/Java, a concrete amalgamation of the concepts with the programming language Java, is discussed. An operational semantics of the resulting technology is described as a transformation of NSP/Java technology to Java Server Pages technology. Furthermore JSPick is presented, a reverse engineering tool for Java Server Pages based presentation layers. A formal semantics of this CASE tool is given as pseudo-evaluation.

The server pages approach is part of a proposed holistic approach to modeling and developing form based, submit/response style systems. A case study exemplifies how the impedance mismatch between modeling and implementation is mitigated by the approach.



# Contents

<b>Abstract</b>	<b>i</b>
<b>1 Introduction</b>	<b>1</b>
<b>2 Next Server Pages Preliminaries</b>	<b>7</b>
2.1 A Motivating Example . . . . .	7
2.2 The NSP Document Structure . . . . .	9
2.3 NSP Interaction Controls . . . . .	13
<b>3 NSP Coding Guidelines</b>	<b>19</b>
3.1 The Object of Parameter Guidelines . . . . .	19
3.2 Basic Parameter Guidelines . . . . .	21
3.3 NSP Parameter Rules . . . . .	23
3.4 Parameter Guidelines for Arrays . . . . .	28
3.5 Parameter Guidelines for Form Messages . . . . .	30
3.6 Document Structure Guidelines and Rules . . . . .	35
<b>4 Web Presentation Layer Architecture</b>	<b>39</b>
4.1 Model 2 Architecture . . . . .	39
4.2 NSP Functional Decomposition . . . . .	43
4.3 Higher Order Server Pages . . . . .	46
<b>5 Operational Semantics of NSP</b>	<b>49</b>
<b>6 Java Server Pages Reverse Engineering</b>	<b>53</b>
6.1 JSPick - A JSP Design Recovery Tool . . . . .	53
6.2 Formal Semantics of JSPick . . . . .	58
<b>7 Formal Definition of the NSP Type System</b>	<b>65</b>
7.1 Core NSP Grammar . . . . .	66
7.2 Core NSP Type System Strength . . . . .	68
7.3 Core NSP Types . . . . .	71
7.4 Type Operators . . . . .	78
7.5 Environments and Judgments . . . . .	80
7.6 Typing Rules . . . . .	81

<b>8 Conclusion</b>	<b>89</b>
8.1 Related Work . . . . .	89
8.2 Further Work . . . . .	93
8.3 Summary . . . . .	94
<b>A Integration with Form-Oriented Analysis</b>	<b>95</b>
A.1 Problem Description . . . . .	96
A.2 Mapping A Form-Oriented Specification to NSP . . . . .	99
<b>B NSP Language Definition</b>	<b>105</b>
B.1 Context Free Grammar . . . . .	105
B.2 Types . . . . .	108
B.3 Subtyping Relation . . . . .	109
B.4 Type Operators . . . . .	110
B.5 Environments and Judgements . . . . .	110
B.6 Typing Rules . . . . .	111
B.7 Example Type Derivation I . . . . .	115
B.8 Example Type Derivation II . . . . .	118
<b>C NSP Document Type Definitions</b>	<b>121</b>
C.1 NSP Core Language XML DTD . . . . .	121
C.2 NSP Core UID Language SGML DTD . . . . .	123
<b>D Notation</b>	<b>125</b>
<b>E Anlagen gemäß Promotionsordnung</b>	<b>127</b>

# List of Figures

- 1.1 CPDS and CPTS. The figure shows the interplay of server side scripts and the browser and visualizes the notions of client page description safety (CPDS) and client page type safety (CPTS). . . . . 3
  
- 3.1 Example Form Message Type. The type defined in this figure by a class diagram is used in order to explain the NSP parameter guidelines for form message types. . . . . 31
- 3.2 Example Cyclic Form Message Type. The type defined in this figure is used to explain the NSP support for cyclic user defined data in forms . . . . . 34
- 3.3 Example Complex Form Message Type. The type defined in this figure is used to explain the NSP mechanism for putting constraints on the attributes of user defined data. . . . . 35
  
- 4.1 Model 2 Architecture. The figure visualizes the "redirecting request" application model coined Model 2 architecture. The model has become commonly known as following the Model View Controller paradigm. . . . . 40
- 4.2 Model 2 Architecture versus NSP Functional Decomposition. The figure shows a typical control and data flow in a Model 2 architecture system up to details of request dispatching and the improvement of a counterpart system build on Next Server Pages technology by an interaction diagram. . . . . 42
- 4.3 Example Interaction Diagram. The figure shows the login dialogue of a web based mail account. The user logs in and views her inbox. If she stores her password, for a certain time no login is necessary. . . . . 45
- 4.4 Example Form Chart Diagram. The user dialogue given by a form chart in this figure poses a typical design problem that can be given a reusable, flexible solution based on a higher order server page concept. . . . . 47

6.1	JSPick Screenshot. The figure shows a partial type description window of the reverse engineering CASE tool JSPick. Such a window displays type information about the web signature of a Java Server Page and the actual superparameters of its possibly several contained forms. . . . .	55
6.2	JSPick Pseudo-Evaluation. The figure contains the complete specification of the semantics of the reverse engineering tool JSPick with respect to type inference of actual form superparameters. JSPick is a design recovery tool for Java Server Pages based presentation layers. . . . .	63
7.1	UML list definitions. The figure shows two alternative cyclic data models of the list data type. . . . .	76
A.1	Form Chart Diagram. The figure visualizes a task manager feature of a combined CSCW/project management tool. . . . .	98
A.2	Layered Data Model. The figure visualizes the form-oriented data model which underlies a task manager feature of a combined CSCW/project management tool. The data model consists of two layers, a data dictionary and a semantic data model. Data model types are used as opaque reference types in the data dictionary. . . . .	98



# List of Code Examples

2.1	Java Server Pages: counter example - form definition . . . . .	8
2.2	Java Server Pages: counter example - targeted page . . . . .	9
2.3	Java Server Pages: counter example - generated client page . . . .	10
2.4	NSP: form definition . . . . .	11
2.5	NSP: targeted page . . . . .	12
2.6	NSP: hyperlink . . . . .	13
3.1	Java: well typed code fragment . . . . .	20
3.2	NSP: non well typed code fragment . . . . .	20
3.3	NSP: form fragment . . . . .	27
3.4	NSP pseudo code: form fragment . . . . .	27
3.5	NSP: gathering data for an array parameter . . . . .	29
3.6	NSP: gathering data for an indexed array parameter . . . . .	29
3.7	NSP: gathering form message type data . . . . .	32
3.8	NSP: gathering distributed form message type data . . . . .	33
3.9	NSP: gathering data for dynamic data structures . . . . .	34
3.10	NSP: constraints on form message type attributes . . . . .	36
4.1	NSP: functional decomposition with dialogue submethods . . . . .	44
4.2	NSP: higher order server pages . . . . .	48
6.1	Java Server Pages: example input to the JSPick CASE tool . . . .	54
7.1	Core NSP: combining static and dynamic document parts . . . . .	69
7.2	Core NSP: gathering data of recursive type . . . . .	75
A.1	Dialogue Constraint Language: task managing feature . . . . .	99
A.2	NSP: task managing feature - task manager . . . . .	101
A.3	NSP: task managing feature - remove task . . . . .	101
A.4	NSP: task managing feature - edit task . . . . .	102
A.5	NSP: task managing feature - change task . . . . .	103
A.6	NSP: task managing feature - delete task . . . . .	103
B.1	Core NSP - derivation with higher order pages . . . . .	115
B.2	Core NSP - derivation of actual parameter . . . . .	118
B.3	Core NSP - targeted Core NSP server page . . . . .	119

