The subject of this dissertation is the construction of two taxonomies and a toolkit of algorithms solving two different but closely related problems from the domain of regular tree languages.

In our context, a taxonomy is a classification of algorithms according to their essential details. Its starting point is a high-level algorithm whose correctness is easily shown. By adding details, one obtains refinements or variations of that algorithm and of algorithms obtained in such a way. This may lead to algorithms from the literature or to new ones. Every detail has correctness arguments associated with it, so that each algorithm's correctness follows from the correctness of the starting point and details added. The sequence of details from the starting point to an algorithm shows how the algorithm can be obtained from the starting point and can be used to characterize the particular algorithm. An algorithm taxonomy may be depicted as a directed acyclic graph, in which nodes refer to algorithms and branches refer to details.

We construct taxonomies for two algorithmic problems from the domain of regular tree languages on ordered, ranked trees. This domain has a rich theory, and parts of this theory have broad applicability in areas such as code generation in compilers—particularly for instruction selection or optimization—and term rewriting. Underlying the practical applications are a number of algorithmic problems, of which tree acceptance and tree pattern matching are two. Many algorithms solving these problems have been described and used in practice. Unfortunately, the domain suffers from a number of deficiencies:

1. Inaccessibility of the theory and algorithms, as they are scattered over the literature and few or no overview publications—particularly algorithm oriented ones—exist.
2. Difficulty of comparing the algorithms due to differences in presentation style and level of formality.
4. Lack of a large and coherent collection of implementations of the algorithms.
5. Difficulty of choosing between different algorithms for practical applications.

The construction of taxonomies aims to solve the first three deficiencies. The construction process is part of TABASCO, a method for domain modeling and domain engineering for a particular kind of domain in computer science. As a first step, an overview of relevant parts of the theory is created. A literature survey is then performed to gather algorithms solving the algorithmic problem or problems. Based on the results, the algorithms are rephrased in a common presentation style and a model in the form of an algorithm taxonomy for the algorithmic problem or problems is constructed. The taxonomy makes the commonalities and variations between the algorithms as well as the correctness arguments for the algorithms explicit.

The problems of tree acceptance and tree pattern matching are related and the algorithms solving them involve many of the same algorithmic ingredients. The commonalities lead to similarities between the taxonomies constructed for each of the two problems. Furthermore, references to and discussions of the algorithms' occurrences in the literature are included.

The last two of the five deficiencies mentioned above can be solved using TABASCO's domain engineering steps, which include the creation of a toolkit of algorithm implementations and of a domain specific language to allow more effective use of such a toolkit. The design of a coherent toolkit is simplified by the availability of a taxonomy, which indicates the commonalities and differences between the various algorithms. The high-level design choices are guided by the taxonomy structure, and the choice of which language constructs to use to implement various design parts can be made based on standard design techniques.

We consider the domain of keyword pattern matching as a brief case study for all of the method's domain engineering steps. For the domain of regular tree languages, we consider the design of a toolkit of tree acceptance and tree pattern matching algorithms based on the two taxonomies. The taxonomy-based toolkit design results in a coherent and easily extendable toolkit.