4 Analysis of Security-Relevant Semantics of BPEL

In this section, the results of an analysis of BPEL as a specification language will be presented. The purpose of this analysis was to identify the intrinsic potential of BPEL to specify business process behaviour that is able to violate restrictions imposed by security policies. These results form the basis for the methods and procedures for security policy assessment introduced in the next chapter. Readers familiar with syntax and semantics of BPEL as a specification language may skip the overview in Section 4.2.

Here and in the following chapters, the term “security relevance” when used with respect to particular elements of a programming or scripting language (e.g., BPEL) or to combinations of such elements denotes the potential of these elements or combinations thereof to specify behaviour of a program (or a business process in the case of BPEL) that could violate a security policy-implied restriction. Once the security-relevant semantics of BPEL have been identified, the assessment of compliance between a BPEL-defined business process and a particular security policy will become more operable since checking the BPEL script for compliance can be restricted to looking for security-relevant semantics contained therein.

4.1 Scope of Analysis

For the sake of general applicability, the aim is to analyse security relevance of semantics expressible via BPEL as a specification language as broadly as possible, without referring to specific types of application to which a particular BPEL script-defined business process may be related. For this purpose, the smallest possible pieces of BPEL scripts had to be identified for which assertions with respect to security relevance could sensibly be made. This cannot be the language constructs of BPEL on their own because, similar to many other programming languages or scripting languages, most or even all of the individual language constructs do not present any security relevance or the security relevance cannot be determined at this “atomic” level.

As can be easily seen, for instance, the security relevance of an assignment operator or command (as present in nearly all programming or scripting languages) cannot be determined if considered in isolation since an assignment operation simply copies values from one location to another usually within registers or main memory of a computer. In order to decide the security relevance of such an operation, further information on the values and the locations involved in this operation is required such as what kind of values are to be copied (e.g., values representing security-classified information) and what operation are to be performed to the
40 Analysis of Security-Relevant Semantics of BPEL

target location during the further processing of the program (e.g., stored on an externally accessible storage or displayed to a user on a terminal screen). If considered together with the restrictions that could be imposed to the values and locations involved, the security relevance of an assignment operation becomes easier to be investigated, but still it is not generally decidable for all types of restrictions whether an operation is secure or not.

The whole of a specific program and even the environment in which the program will be executed may have to be considered to decide this. For instance, if the values copied represent information that is restricted by the security policy to not be leaked to users of the program unless they possess a special authorisation to access this information and the target location would be known to get displayed on a user terminal in the further processing of the program, also knowledge about the environment in which the program will be executed would be required to decide whether the assignment operation violates the security policy or not. If the environment assures that only appropriately authorised persons have access to the room where the particular terminal is installed then the operation may be considered secure. If, however, also people without such authorisation may watch the screen where this information is going to be displayed, that is the environment does not provide measures to avoid this, then the assignment operation would have to be considered insecure.

Admittedly, approaches to security analysis of a programming language usually do not claim to be so far-reaching as to also include considerations concerning the prospective environmental conditions of program execution, but are based on implicit or explicit expectations that these conditions will be appropriate to support security requirements, for example, to prevent inadvertently leakage of information to observers not adequately authorised. Also in this analysis, considerations with respect to environmental conditions as mentioned above will be out of scope. As usual in this type of analyses it is anticipated that appropriately secure environmental conditions are provided by the organisation running the platform on which a business process is to be executed. Since herein a specification language at business process level is subject to the analysis, which is unable to directly present any information to users (this could, at most, be achieved indirectly via messages sent to a Web service or to an invoker of a business process, which usually is also a program), this approach is particularly justified.

4.1.1 Search for Security-Relevant Building Blocks of BPEL Semantics

From the above considerations, combinations of the language constructs with typical restrictions implied by security policies were deemed to be suitable as the smallest parts of BPEL scripts for which sensible statements with respect to their security relevance could be established. Before examining the semantics of these combinations, restrictions derived from security policies for BPEL-defined business processes have been investigated. From this investigation, categories or classes of