(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
2 October 2003 (02.10.2003)

PCT

(10) International Publication Number
WO 03/081457 A1

(51) International Patent Classification[7]: G06F 17/21

(21) International Application Number: PCT/US03/06903

(22) International Filing Date: 5 March 2003 (05.03.2003)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
10/102,189 19 March 2002 (19.03.2002) US

(71) Applicant: **LOGICLIBRARY, INC.** [US/US]; 2717 Hwy 14 W., Suite M, Rochester, MN 55901 (US).

(72) Inventors: **CARLSON, Brent, A.**; 407 14th Ave. SW, Rochester, MN 55902 (US). **GROSS, Dave, H.**; 5211 Palmer Lane NW, Rochester, MN 55901 (US). **GRASER, Timothy, J.**; 1317 38th St. NW, Rochester, MN 55901 (US).

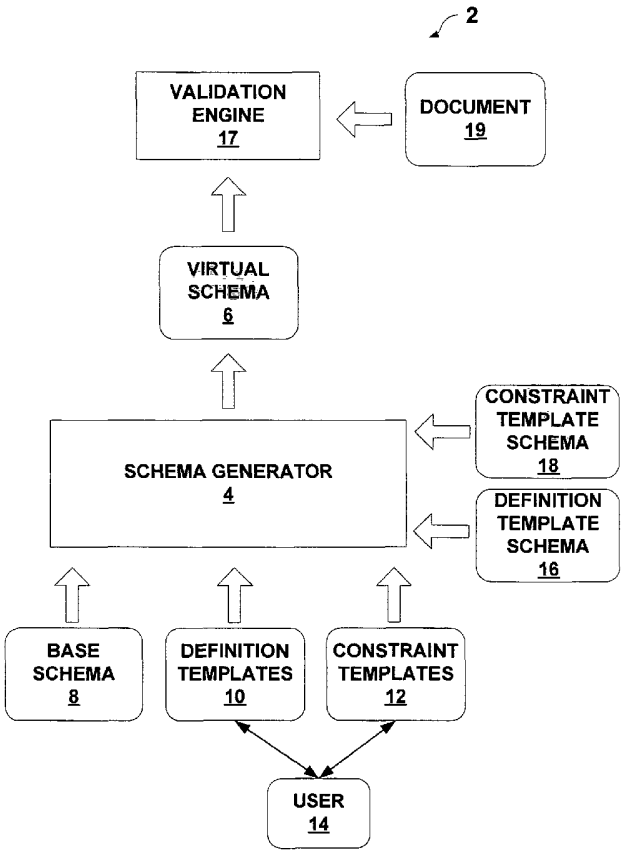
(74) Agent: **SIEFFERT, Kent, J.**; Shumaker & Sieffert, P.A., 8425 Seasons Parkway, Suite 105, St. Paul, MN 55125 (US).

(81) Designated States *(national)*: AE, AG, AL, AM, AT, AU, AZ, BA, BB, BG, BR, BY, BZ, CA, CH, CN, CO, CR, CU, CZ, DE, DK, DM, DZ, EC, EE, ES, FI, GB, GD, GE, GH, GM, HR, HU, ID, IL, IN, IS, JP, KE, KG, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, MA, MD, MG, MK, MN, MW, MX, MZ, NO, NZ, OM, PH, PL, PT, RO, RU, SC, SD, SE, SG, SK, SL, TJ, TM, TN, TR, TT, TZ, UA, UG, UZ, VC, VN, YU, ZA, ZM, ZW.

(84) Designated States *(regional)*: ARIPO patent (GH, GM, KE, LS, MW, MZ, SD, SL, SZ, TZ, UG, ZM, ZW), Eurasian patent (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European patent (AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HU, IE, IT, LU, MC, NL, PT, RO, SE, SI, SK, TR), OAPI patent (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, ML, MR, NE, SN, TD, TG).

[Continued on next page]

(54) Title: DYNAMIC GENERATION OF SCHEMA INFORMATION FOR DATA DESCRIPTION LANGUAGES



(57) Abstract: In general, techniques are described for dynamically generating schema information for data description languages. The techniques include receiving a schema (8) defining a class of elements that conform to a data description language, and receiving a constraint template (18) that defines requirements for instances of the elements. The techniques further comprise dynamically generating schema information based on the schema (8) and the constraint template (18), and validating a document in accordance with the schema (8) information.

WO 03/081457 A1

Published:
— *with international search report*

For two-letter codes and other abbreviations, refer to the "Guidance Notes on Codes and Abbreviations" appearing at the beginning of each regular issue of the PCT Gazette.

DYNAMIC GENERATION OF SCHEMA INFORMATION FOR DATA DESCRIPTION LANGUAGES

TECHNICAL FIELD

[0001] The invention relates to computing devices and, more particularly, to data description languages, such as the extensible markup language (XML), that are often used by computing devices when communicating data

BACKGROUND

[0002] The extensible markup language (XML) has recently emerged as a powerful language for describing and communicating data. In particular, XML is an open, text-based markup language that provides structural and semantic information to the data. XML is a subset of the popular Standard Generalized Markup Language (SGML), and has become widely used within the Internet.

[0003] An XML document includes a root element and, possibly, a number of child elements. Each element consists of an opening tag and a closing tag. The elements of a document must be nested in the closing tag for the child. In this manner, an XML document follows a tree structure in which the elements have parent-child relationships. For example, the following pseudocode illustrates the format of a conventional XML document:

```
<ROOT>
    <CHILD A>
        <CHILD A1> DATA </CHILD A1>
    </CHILD A>
    <CHILD B> DATA </CHILD B>
</ROOT>
```

[0004] An XML schema is used to define and describe a class of XML documents. More specifically, an XML schema uses schema components to define the meaning, usage and relationships of the elements that may be used within a class of XML documents, as well as permissible content, attributes, and values for the elements. The World Wide Web Consortium (W3C) XML Schema Definition Language, for example, is an XML language for describing and constraining the content of XML documents. Other example schema definition languages include Document Content Description for XML (DCD), Schema for

Object-Oriented (SOX), Document Definition Markup Language (DDML), also referred to as XSchema, Regular Language description for XML Core (RELAX), Tree Regular Expressions for XML (TREX), Schematron (SCH), and Examplotron (EG).

[0005] The following pseudocode illustrates the format of a conventional XML schema:

```
</XSD:SCHEMA>
    <XSD:ELEMENT NAME="USED-CAR" MINOCCURS="0" MAXOCCURS="UNBOUNDED">
       <XSD:ELEMENT NAME="MODEL" TYPE="XSD:STRING" USE="REQUIRED"/>
       <XSD:ELEMENT NAME="YEAR" USE="REQUIRED">
             <XSD:ATTRIBUTE NAME="VALUE" TYPE="XSD:INTEGER"/>
          </XSD:ELEMENT>
    </XSD:ELEMENT>
</XSD:SCHEMA>
```

The above pseudocode illustrates some of the basic concepts supported by schema languages. For example, various elements can be defined, such as the elements USED-CAR, MODEL and YEAR defined above. In addition, basic constraints for the elements can be defined, such as whether the element or an attribute of the element is required, and a range for the number of occurrences for the element.

[0006] However, making use of schema languages to constrain the structure and content of the XML documents can lead to very complex schemas having a specific definition for each permissible element. For example, schema languages tend to require definition of specific elements within the schema in order to define constraints on the elements. This approach tends to cause the compliant XML documents to lose normalization. In other words, this approach can result in XML documents in which the names and attributes for the elements are significantly different.

SUMMARY

[0007] In general, the invention is directed to techniques for dynamically generating schema information for data description languages, such as the extensible markup language (XML). More specifically, the techniques dynamically generate schema information by application of one or more constraint templates to a base schema. One or more definition templates may also be applied to extend the base schema as necessary to further define the valid types of elements. In this manner, complex schema information

can be generated including requirements that constrain the use and relationships of the elements without requiring modification to the base schema.

[0008] In one embodiment, a method comprises receiving a schema defining a class of elements that conform to a data description language, and receiving a constraint template that defines requirements for instances of the elements. The method further comprises validating a document in accordance with the schema and the constraint template.

[0009] In another embodiment, a system comprises a schema defining a class of elements that conform to a data description language, and a constraint template that defines requirements for instances of the elements. The system further comprises a validation engine to validate a document in accordance with the schema and the constraint template.

[0010] In another embodiment, a system comprises a schema defining a class of elements that conform to a data description language, a definition template that defines a sub-class for the elements, and a constraint template that defines requirements for instances of the sub-class. The system further comprises a schema generator to generate a virtual schema based on the schema, the definition template, and the constraint template, and a validation engine to validate a document in accordance with the virtual schema.

[0011] In another embodiment, a computer-readable medium comprises instructions to cause a processor to receive a schema defining a class of elements that conform to a data description language, and receive a constraint template that defines requirements for instances of the elements. The medium further comprises instructions to dynamically generate schema information based on the schema and the constraint template.

[0012] In another embodiment, a system comprises a schema defining a class of elements that conform to a data description language, and a hierarchy of constraint templates having at least one parent constraint template and at least one child constraint template, wherein the constraint templates defines requirements for instances of the elements, and the requirements of the child constraint template limit the requirements of the parent constraint template.

[0013] The techniques described herein may offer one or more advantages. For example, the constraint templates and the definition templates conform to the data description language, e.g., XML, and can easily be created and modified. Accordingly, a user can define constraints for the elements of the schema without needing to modify the schema itself. Furthermore, a constraint template schema and a definition template schema can be used to control the structure and content of the templates. The templates created by the user can, therefore, be validated against their respective schemas to identify any errors

within the templates. In this manner, the techniques reduce the chances of error in that users need not continually create or modify a schema in order to support new classes of XML documents.

[0014] In addition, the XML documents that comply with the base schema are normalized in the sense that the names and attributes of the elements are substantially similar. Accordingly, constraints can be introduced for fine-grain validation of XML documents without requiring a complex schema in which the documents lose their normalization.

[0015] The details of one or more embodiments of the invention are set forth in the accompanying drawings and the description below. Other features, objects, and advantages of the invention will be apparent from the description and drawings, and from the claims.

BRIEF DESCRIPTION OF DRAWINGS

[0016] FIG. 1 is a block diagram illustrating an example system in which a schema generator dynamically generates schema information, referred to as a "virtual" schema.

[0017] FIG. 2 is a block diagram illustrating a hierarchy of constraint templates for use by the schema generator.

[0018] FIG. 3 is a flowchart illustrating in further detail the techniques for dynamically generating the schema information.

[0019] FIG. 4 is a block diagram illustrating a system in which a validation engine uses the virtual schema to drive a data capture process.

DETAILED DESCRIPTION

[0020] FIG. 1 is a block diagram illustrating an example system 2 in which a schema generator 4 dynamically generates schema information, referred to as virtual schema 6. In particular, schema generator 4 generates virtual schema 6 in accordance with base schema 8, definition templates 10, and constraint templates 12.

[0021] Base schema 8 conforms to a schema definition language, and defines a class of elements that conform to a data description language, such as the extensible markup language (XML). In this manner, base schema 8 may remain static and need not be modified to support new XML documents.

[0022] To define classes of permissible XML documents, user 14 may create definition templates 10, constraint templates 12, or both. More specifically, the user may create one or more definition templates 10 that define sub-classes for the elements defined by base schema 8. In this manner, user 14 can extend the element definitions of base schema 8 without modifying base schema 8.

[0023] In addition, user 14 may create one or more constraint templates 12 that define requirements for instances of the elements. Constraint templates 12 may define requirements for instances of elements belonging to the classes defined by base schema 8, instances of elements belonging to the sub-classes defined by definition templates 10, or both. For example, constraint templates 12 may define a required cardinality for the instances of the elements, a required minimum or maximum number of the instances, a range for a required number of the instances of the elements, a required attribute for the instances, a required parameter value for the instances of the elements, specific required instances of the elements, and the like.

[0024] Schema generator 4 generates the schema information of virtual schema 6 by first generating a data structure representing the classes of elements defined by base schema 8. Schema generator 4 then applies definition templates 10 to base schema 8 to extend the schema information to include the sub-classes of elements defined within definition templates 10. Finally, schema generator 4 applies constraint templates 12 to update the schema information to include the restrictions defined by constraint templates 12.

[0025] Definition templates 10 and constraint templates 12 conform to the data description language to which the elements of base schema 8 comply, e.g., XML. Accordingly, user 14 can easily create and modify definition templates 10 and constraint templates 12, and need only modify base schema 8 in order to support new classes of XML documents 19.

[0026] Schema generator 4 validates definition templates 10 and constraint templates 12 created by user 14 against definition template 16 and constraint template schema 18, respectively. In this manner, definition template schema 16 and constraint template schema 18 can be used to validate the structure and content of the templates 10, 12. In this manner, the techniques can reduce the chances of error by user 14.

[0027] Documents 19 represent electronic documents, files, scripts, communication streams, software objects, and the like, that contain metadata conforming to the data description language. More specifically, the metadata includes instances of elements belonging to the classes of elements defined by schema 8, or the sub-classes of elements

defined by definition templates 10. Example languages include the Extensible Markup Language (XML), Extensible Style Language (XSL), Extensible Linking Language (XLL), Standardized Multimedia Authoring Language (SMIL), as well as variations of the Standard Generalized Markup Language (SGML).

[0028] Validation engine 17 validates documents 19 in accordance with virtual schema 6 generated by schema generator 4. In particular, validation engine 17 verifies that the instances of the elements within documents 19 comply with the classes of elements defined by base schema 6, the sub-classes defined by definition templates 10, and the constraints for the instances defined by constraint templates 12.

[0029] Advantageously, documents 19 comply with the base schema and, therefore, are normalized in the sense that the names and attributes of the elements are substantially similar. Accordingly, base schema 8 can be extended, and constraints can be introduced for fine-grain validation of documents 19, without requiring a complex schema in which the documents would otherwise lose normalization.

[0030] The following pseudocode illustrates an exemplary base schema, definition template and constraint template that may be used for capturing information related to reusable software assets. In particular, the following exemplary base schema defines a parent class of elements named ASSET, and two child classes of elements named KEYWORD and RELATION.

```
<XSD:SCHEMA >
   <XSD:ELEMENT NAME="ASSET">
      <XSD:ELEMENT NAME="KEYWORD" MINOCCURS="0" MAXOCCURS="UNBOUNDED">
         <XSD:ATTRIBUTE NAME="NAME" TYPE="XSD:STRING" USE="REQUIRED "/>
         <XSD:ATTRIBUTE NAME="VALUE" TYPE="XSD:STRING" USE="REQUIRED"/>
      </XSD:ELEMENT>
      <XSD:ELEMENT NAME="RELATION" MINOCCURS="0" MAXOCCURS="UNBOUNDED">
         <XSD:ATTRIBUTE NAME="ROLE" TYPE="XSD:STRING" USE="REQUIRED"/>
         <XSD:ATTRIBUTE NAME="ID" TYPE="XSD:ID" USE="REQUIRED"/>
         <XSD:ATTRIBUTE NAME="TYPE" TYPE="XSD:STRING" USE="REQUIRED"/>
      </XSD:ELEMENT>
      <XSD:ATTRIBUTE NAME="NAME" TYPE="XSD:STRING" USE="REQUIRED"/>
      <XSD:ATTRIBUTE NAME="TEMPLATE" TYPE="XSD:STRING" USE="REQUIRED"/>
   </XSD:ELEMENT>
</XSD:SCHEMA>
```

The following exemplary definition template illustrates the definition of sub-classes for the classes of elements KEYWORD and RELATION, thereby extending the definitions provided by the above-listed exemplary base schema.

```
<TEMPLATE NAME="ASSET-DEFINITION-TEMPLATE" PARENT="ASSET-SCHEMA.XSD">
   <DEFINE-KEYWORD NAME="CATEGORY" TYPE="STRING">
      <ADD-VALUE VALUE="FINANCE"/>
      <ADD-VALUE VALUE="BANKING"/>
   </DEFINE-KEYWORD>
   <DEFINE-KEYWORD NAME="PRICE" TYPE="DECIMAL"/>
   <DEFINE-KEYWORD NAME="ALIAS" TYPE="STRING"/>
   <DEFINE-RELATION ROLE="USES" TYPE="ASSOCIATION"/>
   <DEFINE-RELATION ROLE="PREDECESSOR" TYPE="PREVIOUS-VERSION">
      <MAX-OCCURS VALUE="1"/>
   </DEFINE-RELATION>
</TEMPLATE>
```

The above-illustrated exemplary definition template makes use of elements DEFINE-KEYWORD and DEFINE-RELATION to define specific sub-classes for these respective classes of elements defined by the exemplary base schema. More specifically, for class KEYWORD, the exemplary definition template defines a sub-class CATEGORY having two possible values: FINANCE and BANKING. The exemplary definition template defines two additional sub-classes for the class KEYWORD including PRICE and ALIAS. For the class RELATION, the definition template defines two sub-classes of USES and PREDECESSOR.

[0031] The following exemplary constraint template provides requirements for the use of, and constraints for, the instances of the elements.

```
<TEMPLATE NAME="ASSET-CONSTRAINT-TEMPLATE" PARENT="ASSET-DEFINITION-
TEMPLATE.XML">
   <USE-KEYWORD NAME="CATEGORY"/>
   <USE-KEYWORD NAME="PRICE">
      <MAX-OCCURS VALUE="1"/>
   </USE-KEYWORD>
   <USE-RELATION ROLE="PREDECESSOR"/>
   <USE-RELATION ROLE="USES">
      <MIN-OCCURS VALUE="1"/>
   </USE-RELATION>
```

```
</TEMPLATE>
```

The above-illustrated exemplary constraint template makes use of elements USE-KEYWORD and USE-RELATION to define specific requirements for instances for the sub-classes of elements defined by the definition template. More specifically, the exemplary constraint template 10 allows at least one instance of an element belonging to the sub-class CATEGORY. The exemplary constraint template further allows at most one instance of an element belonging to the sub-class PRICE. Similarly, the exemplary constraint template allows at least one instance of an element belonging to the sub-class PREDECESSOR, and requires at least one instance of an element belonging to the sub-class USES.

[0032] The following pseudocode illustrates an exemplary document that describes a reusable software asset, and which complies with the exemplary base schema, definition template, and constraint template listed above.

```
<ASSET NAME="BANKING-ASSET-2.0" TEMPLATE="ASSET-CONSTRAINT-
TEMPLATE.XML">
   <KEYWORD NAME="CATEGORY" VALUE="BANKING"/>
   <KEYWORD NAME="PRICE" VALUE="100.00"/>
   <RELATION ROLE="USES" ID="CURRENCY-ASSET-4.1" TYPE="ASSOCIATION"/>
   <RELATION ROLE="PREDECESSOR" ID="BANKING-ASSET-1.0" TYPE="PREVIOUS-
VERSION"/>
</ASSET>
```

[0033] FIG. 2 is a block diagram illustrating a hierarchy 20 of constraint templates 22A - 22I (herein constraint templates 22). Constraint templates 22 are arranged accordingly to parent / child relationships in which the child constraint templates include requirements that further limit the requirements defined by the parent constraint templates. For example, constraint template 22B is a child of constraint template 22A, and further limits the requirements defined by constraint template 22A. Similarly, constraint template 22D further limits the requirements defined by constraint template 22B.

[0034] To generate virtual schema 6, schema generator 4 can use any node of hierarchy 20, i.e., any of constraint templates 22. If constraint template 22F is used, for example, schema generator 4 applies the ancestor constraint templates 22A, 22B, 22D, and 22F to fully constrain the use of instances of element classes and sub-classes defined by base schema 8 and definition template 10. During the application of the constraint templates

22, schema generator 4 validates the constraint templates 22 against constraint template schema 16. In addition, schema generator 4 validates that each child constraint template applied does not conflict with any ancestor constraint template. For example, if a parent constraint template defines a minimum number of instances for an element, a child constraint template cannot define a lower minimum.

[0035] User 14 can easily configure system 2 to support a new class of XML documents by adding one or more constraint templates 22 to hierarchy 20. In this manner, constraint templates can readily be developed to support an entire enterprise by developing the templates in hierarchical fashion. Advantageously, all of the XML documents supported by hierarchy 20 conform to base schema 8, and are normalized in that the names and attributes of the elements are substantially similar.

[0036] FIG. 3 is a flowchart illustrating in further detail the techniques for dynamically generating schema information. Initially, schema generator 4 receives a base schema 8 defining a class of elements that conform to a data description language (30). Next, schema generator 4 may receive any number of definition templates that define sub-classes for the elements (32). Upon receiving definition templates 10, schema generator 4 validates the definition templates using definition template schema 16 (36).

[0037] If schema generator 4 determines that any of the definition templates are not valid in view of definition template schema 16 (no branch of 36), schema generator 4 displays an error message to user 14 (50).

[0038] Otherwise, schema generator 4 receives any number of constraint templates 12 that define requirements for instances of the elements (38). As described above, the constraint templates 12 may be associated in hierarchical form. Upon receiving the constraint templates 12, schema generator 4 validates each constraint template using constraint template schema 18 (40). If schema generator 4 determines that any of the constraint templates 12 are invalid in view of constraint template schema 18 (no branch of 36), schema generator 4 displays an error message to user 14 (50).

[0039] Upon validating both definition template 10 and constraint template 12, schema generator 4 generates schema information, i.e., virtual schema 6 (44). In particular, schema generator 4 generates the schema information of virtual schema 6 by first generating a data structure representing the classes of elements defined by base schema 8. Schema generator 4 then applies definition templates 10 to base schema 8 to extend the schema information to include the sub-classes of elements defined within definition template 10. Finally, schema generator 4 applies the constraint templates 12 to update the

schema information to include the restrictions defined by constraint templates 12. As described above, schema generator 4 may apply a hierarchy of constraint templates starting at a root constraint template and proceeding to a selected one of the constraint templates of the hierarchy.

[0040] In one embodiment, schema generator 4 generates virtual schema 6 as a data structure that defines permissible values and attributes for validating the instances of the classes and sub-classes. For every permissible element, the data structure generates a record that defines a name of the element, a list of valid values, a cardinality for the element, and a list of permissible attributes. The data structure further generates a record for each attribute that lists valid values for the attributes. Schema generator 4 may generate the data structure as a tree, hash table, linked list, and the like, or any combination thereof.

[0041] Validation engine 17 validates document 19 in accordance with virtual schema 6 (48). In other words, validation engine 17 verifies that the instances of the elements comply with the classes and sub-classes defined by base schema 8 and definition template 10, respectively, and the requirements defined by constraint templates 13. Similarly, if definition templates 10 are not provided, validation engine validates document 19 in accordance with virtual schema 6 generated from base schema 8 and constraint templates 12. If validation engine 17 determines that document 19 is invalid, validation engine 17 may display an error message to user 14 (50).

[0042] FIG. 4 is a block diagram illustrating a system 60 in which validation engine 17 uses virtual schema 6 to identify any of the required instances of the elements that are missing from document 19. Upon detecting missing instances, validation engine 4 invokes capture module 62 to capture data from the user 14 or repository 64. In this manner, validation engine 17 and capture module 62 may use the base schema 8, definition templates 10, and constraint templates 12 to drive a process for capturing and normalizing data, such as reusable software assets as illustrated in the examples above.

[0043] For example, the captured reusable software assets may comprise a set of related artifacts that have been created for the purpose of applying that asset repeatedly in subsequent development environments. Examples of artifacts include source code or binary code for the software asset. Other examples include documentation such as a requirements specification, design document, and the like. Additional examples of artifacts include independently deployable software components, such as Enterprise JavaBeans (EJB) and the Component Object Model (COM), schemas that conform to a

data description language, modeling information (models) that provides formal representations of various software components, artifacts related to the deployment of the asset, such as particular instances of the software components.

[0044] Various embodiments of the invention have been described. These and other embodiments are within the scope of the following claims.

CLAIMS:

1. A method comprising:

receiving a schema defining a class of elements that conform to a data description language;

receiving a constraint template that defines requirements for instances of the elements; and

validating a document in accordance with the schema and the constraint template.

2. The method of claim 1, wherein validating a document comprises dynamically generating schema information based on the schema and the constraint template for validating the document.

3. The method of claim 1, wherein validating a document comprises:

dynamically generating a virtual schema having a data structure to store schema information representing the schema;

updating the data structure based on the requirements defined by the constraint template; and

validating instances of elements within the document in accordance with the classes defined by the schema and the requirements defined by the constraint template.

4. The method of claim 1, wherein the constraint template defines a required cardinality for the instances of the elements.

5. The method of claim 1, wherein the constraint template defines one of a required minimum and a required maximum number of the instances of the elements.

6. The method of claim 1, wherein the constraint template defines a range for a required number of the instances of the elements.

7. The method of claim 1, wherein the constraint template defines a required attribute for the instances of the elements.

8. The method of claim 1, wherein the constraint template defines a required parameter value for the instances of the elements.

9. The method of claim 1, wherein the constraint template defines specific required instances of the elements.

10. The method of claim 9, wherein validating a document comprises identifying any of the required instances of the elements that are missing from the document.

11. The method of claim 10, further comprising presenting an interface to a user to capture data for the missing required instances.

12. The method of claim 1, wherein the constraint template conforms to the data description language.

13. The method of claim 1, wherein the data description language comprises the extensible markup language (XML).

14. The method of claim 1, further comprising validating the constraint template according to a constraint template schema.

15. The method of claim 1, further comprising generating the constraint template based on input from a user.

16. The method of claim 1, further comprising receiving a definition template that defines a sub-class for the elements.

17. The method of claim 16, further comprising validating the definition template according to a definition template schema.

18. The method of claim 16, further comprising generating the definition template based on input from a user.

19 The method of claim 1, wherein receiving a constraint template comprises receiving a hierarchy of receiving a constraint templates, and wherein validating a document comprises validating a document in accordance with the hierarchy of constraint templates.

20. A system comprising:

a schema defining a class of elements that conform to a data description language;

a constraint template that defines requirements for instances of the elements; and

a validation engine to validate a document in accordance with the schema and the constraint template.

21. The system of claim 20, further comprising a schema generator to generate a virtual schema based on the schema and the constraint template, wherein the validation engine validates the document with the virtual schema.

22. The system of claim 20, wherein the schema generator dynamically generates a data structure to store schema information representing the schema; and updates the data structure to store additional schema information to represent the requirements defined by the constraint template.

23. The system of claim 20, wherein the constraint template defines a required cardinality for the instances of the elements.

24. The system of claim 20, wherein the constraint template defines one of a required minimum and a required maximum number of the instances of the elements.

25. The system of claim 20, wherein the constraint template defines a range for a required number of the instances of the elements.

26. The system of claim 20, wherein the constraint template defines a required attribute for the instances of the elements.

27. The system of claim 20, wherein the constraint template defines a required parameter value for the instances of the elements.

28. The system of claim 20, wherein the constraint template defines specific required instances of the elements.

29. The system of claim 28, wherein the validation engine identifies any of the required instances of the elements that are missing from the document.

30. The system of claim 29, further comprising a capture module to capture data for the missing required instances.

31. The system of claim 20, wherein the constraint template conforms to the data description language.

32. The system of claim 20, wherein the data description language comprises the extensible markup language (XML).

33. The system of claim 20, wherein the schema generator validates the constraint template according to a constraint template schema.

34. The system of claim 20, further comprising a definition template that defines a sub-class for the elements, wherein the validation engine validates the document in accordance with the definition template.

35. The system of claim 34, further comprising a schema generator to generate a virtual schema based on the schema, the constraint template, and the definition template, wherein the validation engine validates the document with the virtual schema

36. The system of claim 35, wherein the schema generator validates the definition template according to a definition template schema.

37 The system of claim 20, further comprising a hierarchy of constraint templates having at least one parent constraint template and at least one child constraint template, wherein the child constraint template includes requirements that further limits

requirements defined by the parent constraint template, and wherein the validation engine validates the document in accordance with the child constraint template.

38. A system comprising:

a schema defining a class of elements that conform to a data description language;

a definition template that defines a sub-class for the elements;

a constraint template that defines requirements for instances of the sub-class;

a schema generator to generate a virtual schema based on the schema, the definition template, and the constraint template; and

a validation engine to validate a document in accordance with the virtual schema.

39. The system of claim 38, wherein the virtual schema comprises a data structure to store schema information representing the schema, the sub-class, and the requirements.

40. The system of claim 39, wherein the schema generator dynamically generates the data structure.

41. The system of claim 38, wherein the constraint template and the definition template conform to the data description language.

42. The system of claim 38, wherein the constraint template defines specific required instances of the elements, and the validation engine identifies any of the required instances of the elements that are missing from the document.

43. The system of claim 42, further comprising a capture module to capture data for the missing required instances.

44. The system of claim 38, wherein the constraint template defines at least one of:

a required cardinality for the instances of the elements,

a required minimum and a required maximum number of the instances of the elements,

a range for a required number of the instances of the elements,

a required attribute for the instances of the elements, and

a required parameter value for the instances of the elements.

45. The system of claim 38, wherein the constraint template conforms to the data description language.

46. A computer-readable medium comprising instructions to cause a processor to perform the methods of claims 1-19.

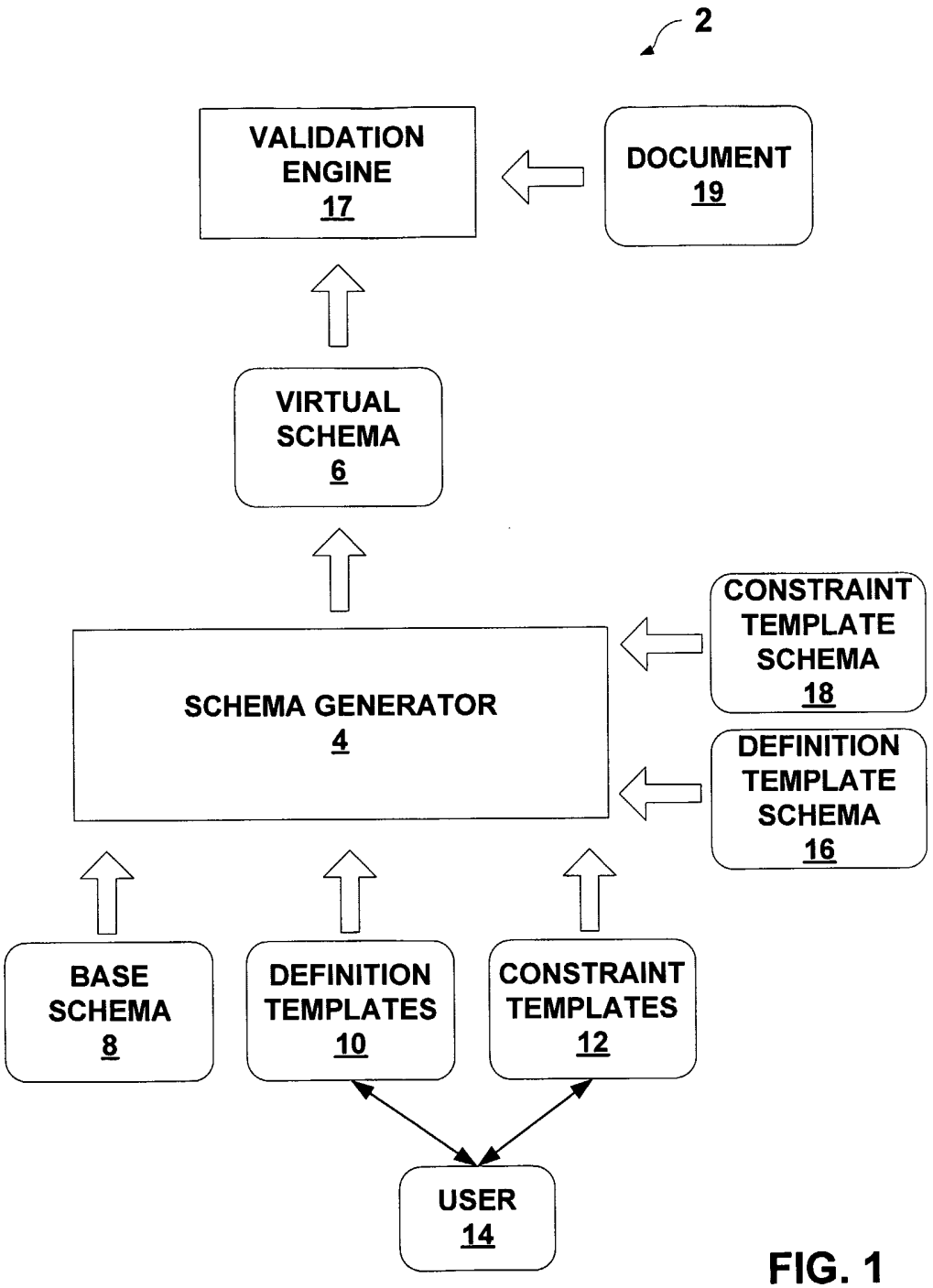

2
VALIDATION ENGINE 17
DOCUMENT 19
VIRTUAL SCHEMA 6
CONSTRAINT TEMPLATE SCHEMA 18
SCHEMA GENERATOR 4
DEFINITION TEMPLATE SCHEMA 16
BASE SCHEMA 8
DEFINITION TEMPLATES 10
CONSTRAINT TEMPLATES 12
USER 14

FIG. 1

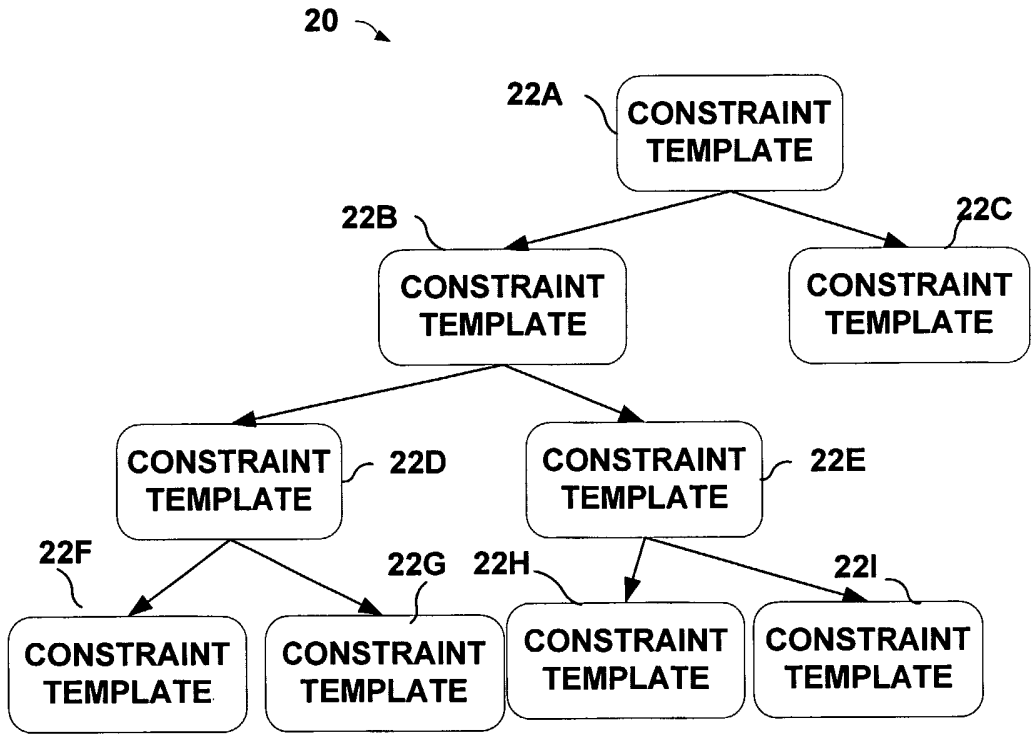

20
22A
CONSTRAINT TEMPLATE
22B
22C
CONSTRAINT TEMPLATE
CONSTRAINT TEMPLATE
CONSTRAINT TEMPLATE
22D
CONSTRAINT TEMPLATE
22E
22F
22G
22H
22I
CONSTRAINT TEMPLATE
CONSTRAINT TEMPLATE
CONSTRAINT TEMPLATE
CONSTRAINT TEMPLATE

FIG. 2

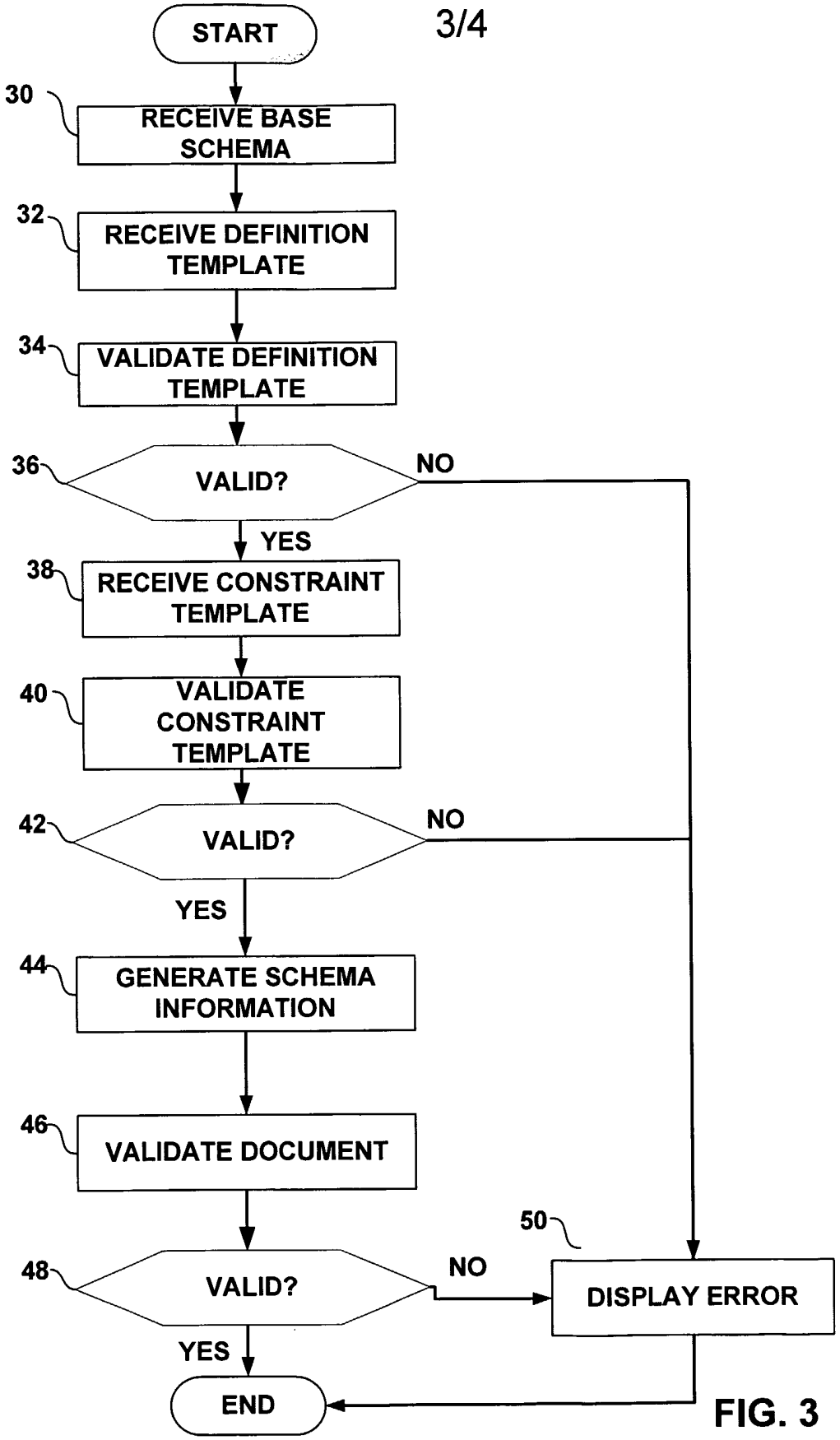

START
30
RECEIVE BASE SCHEMA
32
RECEIVE DEFINITION TEMPLATE
34
VALIDATE DEFINITION TEMPLATE
36
VALID?
NO
YES
38
RECEIVE CONSTRAINT TEMPLATE
40
VALIDATE CONSTRAINT TEMPLATE
42
VALID?
NO
YES
44
GENERATE SCHEMA INFORMATION
46
VALIDATE DOCUMENT
48
VALID?
NO
YES
50
DISPLAY ERROR
END

FIG. 3

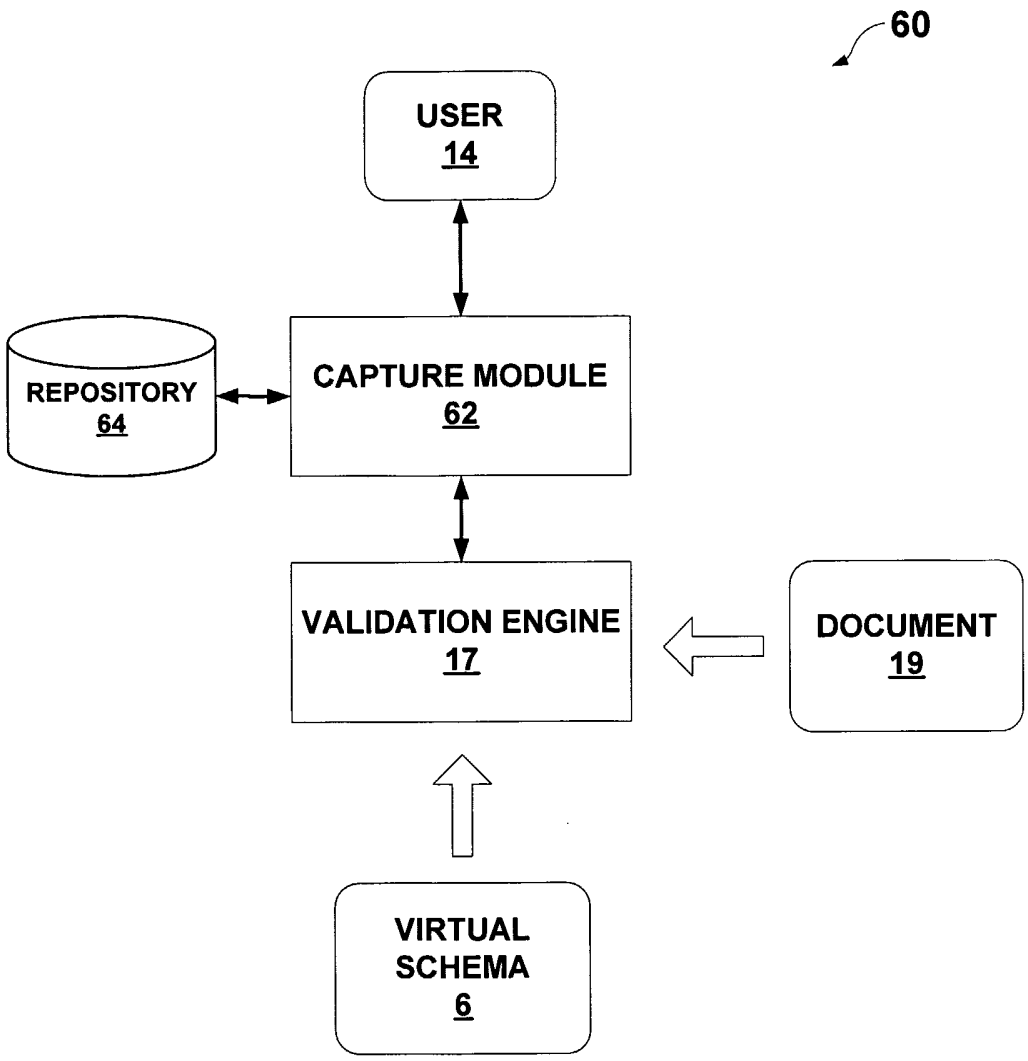

60
USER
14
REPOSITORY
64
CAPTURE MODULE
62
VALIDATION ENGINE
17
DOCUMENT
19
VIRTUAL
SCHEMA
6

FIG. 4

INTERNATIONAL SEARCH REPORT

International application No.
PCT/US03/06903

A. CLASSIFICATION OF SUBJECT MATTER

IPC(7) : G06F 17/21
US CL : 715/513

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)
U.S. : 715/513

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category *	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	US 6,446,256 B1 (HYMAN et al) 03 September 2002 (3.9.2002)	1-45
A	US 6,289,501 B1 (MUTSCHLER, III) 11 September 2001 (11.9.2001)	1-45
A	US 2001/0029604 A1 (DREYBAND et al) 11 October 2001 (11.10.2001)	1-45
A	US 6,154,738 A (CALL) 28 November 2000 (28.11.2000)	1-45
A,P	US 2003/0037301 A1 (RISING, III) 20 February 2003 (20.2.2003)	1-45

☐ Further documents are listed in the continuation of Box C. ☐ See patent family annex.

* Special categories of cited documents:

"A" document defining the general state of the art which is not considered to be of particular relevance

"E" earlier application or patent published on or after the international filing date

"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)

"O" document referring to an oral disclosure, use, exhibition or other means

"P" document published prior to the international filing date but later than the priority date claimed

"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention

"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone

"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art

"&" document member of the same patent family

Date of the actual completion of the international search	Date of mailing of the international search report
15 April 2003 (15.04.2003)	02 MAY 2003
Name and mailing address of the ISA/US Commissioner of Patents and Trademarks Box PCT Washington, D.C. 20231 Facsimile No. (703)305-3230	Authorized officer for Heather R Herndon Telephone No. (703) 306-5631

Form PCT/ISA/210 (second sheet) (July 1998)

INTERNATIONAL SEARCH REPORT

International application No.
PCT/US03/06903

Box I Observations where certain claims were found unsearchable (Continuation of Item 1 of first sheet)

This international report has not been established in respect of certain claims under Article 17(2)(a) for the following reasons:

1. ☐ Claim Nos.:
because they relate to subject matter not required to be searched by this Authority, namely:

2. ☐ Claim Nos.:
because they relate to parts of the international application that do not comply with the prescribed requirements to such an extent that no meaningful international search can be carried out, specifically:

3. ☒ Claim Nos.: 46
because they are dependent claims and are not drafted in accordance with the second and third sentences of Rule 6.4(a).

Box II Observations where unity of invention is lacking (Continuation of Item 2 of first sheet)

This International Searching Authority found multiple inventions in this international application, as follows:

1. ☐ As all required additional search fees were timely paid by the applicant, this international search report covers all searchable claims.
2. ☐ As all searchable claims could be searched without effort justifying an additional fee, this Authority did not invite payment of any additional fee.
3. ☐ As only some of the required additional search fees were timely paid by the applicant, this international search report covers only those claims for which fees were paid, specifically claims Nos.:

4. ☐ No required additional search fees were timely paid by the applicant. Consequently, this international search report is restricted to the invention first mentioned in the claims; it is covered by claims Nos.:

Remark on Protest

☐ The additional search fees were accompanied by the applicant's protest.

☐ No protest accompanied the payment of additional search fees.