UDC 004.8

Ia. Vasyliuk, V. Teslyuk, P. Denysyuk
Lviv Politechnic National University
Computer Aided Systems Department

MEMS DATA TRANSFORMING METHOD FOR ONTOLOGICAL MODELS

In the paper the data obtaining method for ontological models is presented.

Key words: microelectromechanical systems (MEMS), computer-aided systems (CAD), ontology.

Викладено метод отримання даних для онтологічних моделей.

Ключові слова: мікроелектромеханічні системи (MEMS), САПР, онтологія.

Introduction

During last decade the systems complexity increases, the economic and time outgoings at their design and manufacturing grow. The development process needs huge knowledge basis from different scientific fields and practical experience in defined subject areas. Ones of most sophisticated issues, with which the project organizations are met, are information exchange, data gathering, data-knowledge persistence about technical devices, systems, and accordingly their further reuse. That is why the design process should be implemented with the computer-aided systems usage [2] which enables to automate some process.

Problem Formulation

Today microelectromechanical systems (MEMS) [1] play crucial role in all human spheres. They enable to decrease costs of technical products productions of economic application, automate manufacturing processes. That is why in automated MEMS design process based upon ontological models the vital issue is the possibility to data obtain for ontologies. In the paper the MEMS data transforming method for ontological models, which is grounded on previously created microelectromechanical systems, composes ontological models. Further based upon them it is possible to modify or synthase new MEMS structures via “case-based reasoning” approach and genetic algorithms.

Algorithm of Data Obtaining for Ontological Models

The developed method of data obtaining for ontological models consists of five steps: 1) data conversion of format *.dwg to *.xml; 2) data validation of created *.xml file basing upon defined DTD (Data Type Definition) and XML Schema [6]; 3) object creation (Java types) grounded on *.xml file data; 4) objects initialization, data filling of their attributes using JAXB technology [7]; 5) ontological concepts-objects mintage with appropriate qualities [2].

Today there are a lot of systems for microelectromechanical systems (AutoCAd, SolidWorks, DataCard etc.) [3]. Every of them enables to present MEMS structure and its information model in *.dwg format. That permits to use microelectromechanical system data for information file (*.xml format) creation about MEMS structure. Embracing such technologies as ACAD DWG to PDF Converter 5.02 and XEROX [5] the data recreation from *.dwg format to file of *.xml format. In such it becomes possible to process obtained data further.

In second step the received data (xml format) are validated through appropriately designed DTD (Data Type Definition) or XML (Extensible Mark-up Language) Schema [6]. DTD enables to determine elements structure and their attributes in document, comments to elements and their attributes in micromechanical systems, namespaces etc. Comparing to DTD XML Schema defines every element (MEMS) type, their hierarchical structure, value of every element except xml document structure determination.

During third and fourth phase of method work the data conversion of xml format, creation and initialization of object and their appropriate attributes via JAXB technology [7] are performed. This library

enables to create hierarchical tree of objects and assign corresponding values according to data of xml file. To wit if an element is a complex type and consists of some attributes than respectively such object and their attributes hierarchy in object-oriented language (in our case it is java) will be created. As result, the object “FlowSensor” will be created with attributes – “model”, “flowRange”, “calibrationGas”, “flowPortType”, “electricalConnection” etc.

```
import java.io.File;

import javax.xml.bind.JAXBContext;
import javax.xml.bind.JAXBException;
import javax.xml.bind.Unmarshaller;

import com.epam.lab.xmlparser.model.*;

public class JaxbParser implements Runnable {
	String fileName;

	public JaxbParser() {

	}

	public JaxbParser(String fileName) {
		this.fileName = fileName;
	}

	private Paper paper;

	public void parse() throws JAXBException {

		JAXBContext jaxbContext = JAXBContext
				.newInstance("com.cad.mems.xmlparser.model");
		Unmarshaller unmarshaller = jaxbContext.createUnmarshaller();
		paper = (Paper) unmarshaller.unmarshal(new File("mechnicalactuator.xml"));
	}

	public MechanicalActuator getMechanicalActuator() {
		return mechanicalActuator;
	}

	@Override
	public void run() {
		try {
			this.parse();
		} catch (JAXBException e) {
			e.printStackTrace();
		}
	}
}
```

Fig.1 Object creation and initialization

The fifth phase of developed algorithm is the conversion of object-oriented language objects and their attributes in ontological concepts-objects, their properties, constraints and relationships between them. The significant difference between objects of object-oriented language and ontological objects with OWL (Ontology Web Language) [8] is the fact that first ones are basing on objects-types principle, the second – upon that objects properties. To this issue should be pay a great attention because of correct objects appropriateness. For conversion of object-oriented language objects to concepts-objects the Elmo [9] technology is used. With assistance of annotations (@rdf, @inverseOf, @disjointWith) the transformation of objects to ontological concepts is performed, the relationships is established between them.

Data Models and results of developed method

During the developed algorithm at every stage some data models are created. At first that are data models of xml format (fig.2)

```
<?xml version="1.0"?>
<!DOCTYPE FlowSensor SYSTEM "FlowSensor.dtd">
<FlowSensor>
  <Model>D6F-01A1-110</Model>
  <FlowRange min=0 max=1>0</FlowRange>
  <CalibrationGas>air</CalibrationGas>
  <FlowPortType join="barb" minOutsideDiameter=7.44
  maxOutsideDiameter=8.6>8.0</FlowPortType>
  <ElectricalConnection>three-pin<ElectricalConnection>
  .
  .
  .
  <CurrentConsumption maxWithNoload=15 minWithVcc=12
  maxWithVcc=24 temp=25>18</CurrentConsumption>
</FlowSensor>
```

Fig. 2 Data model in xml format

During the second phase the data model of xml format is correlated with developed models of DTD and XML Schema. At fig.3 and fig.4 the validation process of these models is viewed:

```
<?xml version="1.0"?>
<!DOCTYPE FlowSensor SYSTEM "FlowSensor.dtd">
<FlowSensor>
  <Model>D6F-01A1-110</Model>
  <FlowRange min=0 max=1>0</FlowRange>
  <CalibrationGas>air</CalibrationGas>
  <FlowPortType join="barb" minOutsideDiameter=7.44
  maxOutsideDiameter=8.6>8.0</FlowPortType>
  <ElectricalConnection>three-pin<ElectricalConnection>
  .
  .
  .
  <CurrentConsumption maxWithNoload=15 minWithVcc=12
  maxWithVcc=24 temp=25>18</CurrentConsumption>
</FlowSensor>
```

```
<!DOCTYPE FLOWSENSOR [

<!ELEMENT MODEL (#PCDATA)>
<!ELEMENT FLOW RANGE (#PCDATA)>
<!ELEMENT CALIBRATION GAS (#PCDATA)>
<!ELEMENT FLOW PORT TYPE (#PCDATA)>
<!ELEMENT ELECTRICAL CONNECTION (#PCDATA)>
<!ELEMENT CURRENT CONSUMPTION (#PCDATA)>
.
.
.
<!ATTLIST FLOW RANGE MIN CDATA #IMPLIED>
<!ATTLIST FLOW RANGE MAX CDATA #IMPLIED>
<!ATTLIST FLOW PORT TYPE MaxOutsideDiameter CDATA #REQUIRED>
<!ATTLIST FLOW PORT TYPE MinOutsideDiameter CDATA #REQUIRED>
<!ATTLIST CURRENT CONSUMPTION MaxWithNoload #REQUIRED>
<!ATTLIST CURRENT CONSUMPTION MaxWithVcc #REQUIRED>
<!ATTLIST CURRENT CONSUMPTION Temp #REQUIRED>


<!ENTITY MODEL "D6F-01A1-110">
<!ENTITY MODEL "D6F-02A1-110">
<!ENTITY COPYRIGHT "Copyright 1998 Vervet Logic Press">
]>
```

Fig. 3 Example of xml file validation via DTD for "flow sensor"

```
<?xml version="1.0"?>
<!DOCTYPE FlowSensor SYSTEM "FlowSensor.dtd">
<FlowSensor>
  <Model>D6F-01A1-110</Model>
  <FlowRange min=0 max=1>0</FlowRange>
  <CalibrationGas>air</CalibrationGas>
  <FlowPortType join="barb" minOutsideDiameter=7.44
  maxOutsideDiameter=8.6>8.0</FlowPortType>
  <ElectricalConnection>three-pin<ElectricalConnection>
  .
  .
  .
  <CurrentConsumption maxWithNoload=15 minWithVcc=12
  maxWithVcc=24 temp=25>18</CurrentConsumption>
</FlowSensor>
```

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xsd:element name="FlowSensor" type="mems"/>
  <xsd:complexType name="FlowSensor">
  <xsd:sequence>
    <xsd:element name="Model" type="xsd:string"/>
    <xsd:element name="FlowRange" type="xsd:integer"/>
    <xsd:element name="CalibrationGas" type="xsd:string"/>
    <xsd:element name="FlowPortType" type="xsd:integer"/>
    <xsd:element name="ElectricalConnection" type="xsd:integer"/>
    ...
    <xsd:element name="CurrentConsumption" type="xsd:integer"/>

</xs:schema>
```

Fig. 4 Example of xml file validation via XML Schema

During third and fourth stages basing upon validated data model of xml format the respective model-type-object "FlowSensor" is created (fig. 5)

```
public class FlowSensor {

    private Integer flowRange;
    private Integer currentConsumption;
    private String model;
    private String calibrationGas;
    public Integer getFlowRange() {
        return flowRange;
    }

    public void setFlowRange(Integer flowRange) {
        this.flowRange = flowRange;
    }

    public Integer getCurrentConsumption() {
        return currentConsumption;
    }

    public void setCurrentConsumption(Integer currentConsumption) {
        this.currentConsumption = currentConsumption;
    }

    public String getModel() {
        return model;
    }

    public void setModel(String model) {
        this.model = model;
    }

    public String getCalibrationGas() {
```

Fig. 5 Model-type-object "FlowModel"

At the last phase the appropriate ontological model is created, which is based upon object-oriented language objects, fig. 6:

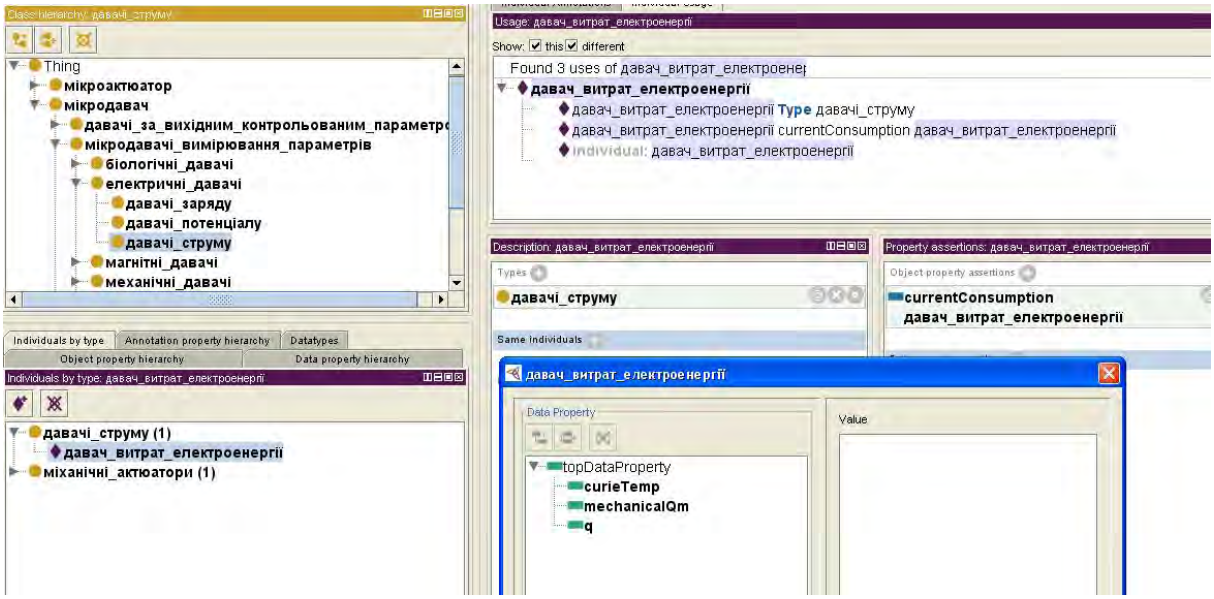

Fig. 6 Ontological model of "FlowSensor" concept

Conclusion

The developed method of data receiving for ontological models enables to converse microelectromechanical systems structure, which is developed by traditional MEMS design tools (AutoCAd, SolidWorks, DataCard etc.), in corresponding concepts-objects of ontological models. So the knowledge fundamentals is forming on the basis of ontological models for various complicated objectives solution which can become during the MEMS design of another types. Respectively, such knowledge can be reused in further multilevel MEMS design or MEMS structure modifications.

1. Теслюк В.М. Моделі інформаційних технологій синтезу мікроелектромеханічних систем: Монографія. – Львів: Вежа і Ко, 2008. – 192 с. 2. Tae-Sul S. Sharin CAD models based on the feature ontology of command history / Yonsook Lee, Sang-Uk Cheon, Lalit Patil // International Journal of CAD/CAM,1598-1800, vol,5, no.1, 2005, – P.47–39. 3. http://en.wikipedia.org/wiki/List_of_computer-aided_design_editors 4. Gallaher M. P. Economic impact assessment of the international standard for the exchange of product model data (STEP) in transportation equipment industry, O'Connor, A. C., and Phelps T., NIST, 2002. – 193 p. 5. http://open.xerox.com/ Services/PDF-to-XML. 6. http://www.w3schools.com/xml/xml_dtd.asp. 7. http://jaxb.java.net/. 8)http://www.w3.org/TR/owl-features/. 9)http://www.openrdf.org/doc/elmo/1.5/usУДК 621.396.6:681.3