RSCDF: A Dynamic and Context-Sensitive Metadata Description Framework for Industrial Resources *

Olena Kaykova, Oleksiy Khriyenko, Anton Naumenko, Vagan Terziyan, Andriy Zharko

Industrial Ontologies Group, Agora Center, University of Jyvaskyla, Finland
e-mail: vagan@it.jyu.fi

Abstract

Emerging Semantic Web technology offered the Resource Description Framework (RDF) as a standard for semantic annotation of Web resources. It is expected that Web content with RDF-based metadata layer and ontological basis for it will be enough to enable interoperable and automated processing of Web data by various applications. However emerging industrial applications consider e.g. machines, processes, personnel, services for condition monitoring, remote diagnostics and maintenance, etc. to be specific classes of Web resources and thus a subject for semantic annotation. Such resources are naturally dynamic, not only from the point of view of changing values for some attributes (state of resource) but also from the point of view of changing "status-labels" (condition of the resource). In this paper we present Resource State/Condition Description Framework (RSCDF), as an extension to RDF, which introduces upper-ontology for describing such characteristics of resources as states and correspondent conditions, dynamics of state changes, target conditions and historical data about previous states. These descriptions are supposed to be used by external Web-services (e.g. condition monitoring, remote diagnostics and predictive maintenance of the resources). We present RSCDF as temporal and contextual extensions of RDF and discuss a State-Symptom-Diagnosis-Decision-Maintenance model as the basis for RSCDF schema. Then we present RSCDF schema itself and show some examples.

1. Introduction

Original idea of Semantic Web as next-generation of the Web assumes that besides existing content there is a conceptual layer of machine-understandable metadata, which makes the content available for processing by intelligent software, allows automatic resource integration and provides interoperability between heterogeneous systems. To be understood by software application, semantics must be presented explicitly in some form, which would allow intelligent information processing substituting human. Such semantic description is a metadata (data about data), attached to a resource. Addressing these problems W3C consortium has started Semantic Web Activity, which resulted in development of *Resource Description Framework* (RDF) as a basic model for semantic descriptions.

The RDF [Klyne & Caroll, 2004, Manola & Miller, 2004] is a system for making statements about Web resources. Each RDF statement is a triple (subject-predicate-object) and may be interpreted as a subject that has a predicate with value object. Also RDF provides a well-defined framework to make statements about statements, which is called reification. According to traditional view a Web resource contains some data (either structured in some or other way or unstructured) about certain domain entity(s). Thus an RDF description is in a way a metadata.

Originally the dynamics of Web resources was not seriously considered as a subject for RDF community consideration. Everything was simple. A human creates Web resource and responsible for its annotation. Resource data is stable for some period of time. When essential changes in data are necessary then the human renews the resource. If changes are such that they effect also the

* Kaykova O., Khriyenko O., Naumenko A., Terziyan V., Zharko A., RSCDF: A Dynamic and Context-Sensitive Metadata Description Framework for Industrial Resources, *Eastern-European Journal of Enterprise Technologies,* Vol. 3, No. 2, 2005, 55-78.

metadata then the human will update also appropriate RDF descriptions. Automated metadata updates are becoming very complicated when we are dealing with dynamic Web resources. In this case resource content is generated automatically according to continuously changing content of some databases and there would be really a challenge to accordingly update the metadata, which summarizes every new state of the resource [Benjamins at al., 1999]. However still these dynamic resources are traditionally informational in a sense that the informational content of original databases is collected by humans and from time to time maintained by humans, which can lead to optimistic assumption that RDF (of course with appropriate ontological support) has enough means already to describe such "discrete" dynamics.

More complicated case is when we are dealing with industrial resources (machines, processes, personnel, etc.), which dynamics is often naturally continuous even if monitored during discrete time intervals. The challenge is to consider such resources as Web resources and thus as subject for semantic annotation as it was discussed in [Kaykova et. al., 2004]. This approach is the part of going-on SmartResource project "Proactive Self-Maintained Resources in Semantic Web" [SmartResource, 2004] leaded by Industrial Ontologies Group [IOG, 2004]. Industrial resources can be linked to the Web by special adapters and sensors. Sensors keep track of data about internal states of the resource and adapters help to make this data readable for external applications. It is obvious that RDF is not very suitable for making statements about continuously changing resources. Another problem is reification, which deals with dynamically changed statements, e.g. descriptions for certain processes or trends taking into account that such descriptions are often context-dependent. In [Kaykova et. al., 2004] it is supposed that semantic descriptions for the industrial resources state and conditions should be generated automatically by appropriate Web-services to be further used when appropriate for predictive maintenance of the original industrial resources.

The goal of this paper is to present *Resource State/Condition Description Framework (RSCDF),* as an extension to RDF, which introduces upper-ontology for describing maintenance-oriented characteristics of resources: states and correspondent conditions, dynamics of state changes that happen, target condition of the resources and historical data about previous states. Resources (e.g. devices) are assumed to have their own state presented as RSCDF descriptions. These descriptions are used by external applications (e.g. remote diagnostics) that support RSCDF and are able to process data presented in such format. Introduction of RSCDF allows solving problems of interoperability and resource heterogeneity. Enabling RSCDF can be considered as an important stage of the SmartResource project activities [SmartResource, 2004].

Very basic view to the role of RSCDF for the SmartResource concept can bee seen in Figure 1.

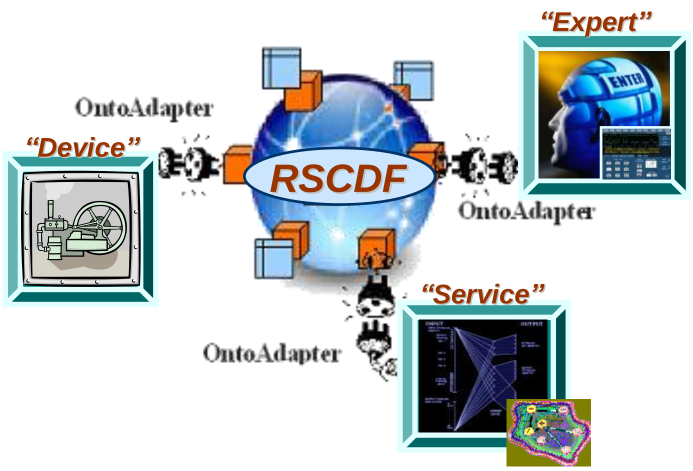

Figure 1 – SmartResource: collaborative environment for field devices, Web-services and human experts with RSCDF-based metadata exchange [SmartResource, 2004]

Data (e.g. diagnostics query) from some field "*Device*" is automatically being translated to RSCDF due to adapter, which is linked to the appropriate ontology. RSCDF will contain ontologically standardized description of the device state (temporal track of the values of parameters taken by sensors) in a form suitable for machine processing and free from the specifics (nature, producer, etc.) of the device itself. After the stage of a suitable "*Expert*" discovery in P2P network of various Web resources the RSCDF query can be shown to the expert in diagnostics (specific adapter will visualize RSCDF to the human) who can return diagnosis (translated automatically back to RSCDF) to the device. Thus device (actually some agent on behalf of the device) will collect some history of device states which is labeled with expert diagnoses. Such history sooner or later will be enough as training set for some machine learning algorithm, which can be represented as Web "*Service*". Thus labeled device state history in RSCDF will be sent to an external machine-learning service; service will build a model (e.g. neuro-fuzzy network); service is ready for remote diagnostics of the device instead of querying expensive experts for the diagnostics. So RSCDF is required to be a semantically rich RDF-based language for enabling interoperability within global systems for automated online condition monitoring and remote diagnostics of heterogeneous field devices.

The following text will be organized as follows. In Chapter 2 we will present RSCDF as temporal and contextual extensions of RDF. Chapter 3 explains supports for data types in RSCDF. In Chapter 4, the State-Symptom-Diagnosis-Decision-Maintenance model will be discussed as the basis for RSCDF schema. Chapter 5 presents the RSCDF schema itself. We conclude in Chapter 6.

2. RSCDF extensions

Resource State/Condition Description Framework (RSCDF) extends RDF to make it expressive enough for the needs of automated resource monitoring and maintenance. Of course we are assuming that there are some Web-accessible industrial resources, which will need monitoring and maintenance. Consider two major extensions: temporal and contextual, which are described in more details below.

2.1 Temporal RSCDF extension

2.1.1 Resource maintenance: a need in temporal metadata

RSCDF ontology, in order to be a data format that supports advanced resource maintenance techniques, must contain temporal concepts as a basis for temporal metadata. Efficient resource maintenance must be based not only on analysis of static resource states, but take into account temporal relations between those states, too. This enables us to have a broader view on the situation: sometimes just considering temporal evolution of relations between different resource states/symptoms can give us a hint as to precise diagnostics. Especially important in this approach is to have ready temporal scenarios of possible critical situations that can occur. Any particular situation can be compared with existing scenarios and possibly classified as belonging to one of them [Ryabov & Terziyan, 2003]. Thus, RSCDF must contain a temporal extension to enable intelligent decision making systems based on temporal diagnostics, i.e. diagnostics based on temporal data. Generally, temporal diagnostics is one important area of application of temporal representation and reasoning formalisms. It includes medical and industrial diagnostics, diagnostics in field device management, etc. [Terziyan & Ryabov, 2003].

Analyzing recent research efforts related to automated monitoring and maintenance of organizations based on multi-agent systems [Jonker et. al., 2001], it becomes clear that automated maintenance of the abstract resource will need means and data models for accumulation of its state history – a sequence of resource states over a time. Thus, it is more correct to describe an object by its history, than by its current state.

To summarize, the three main possibilities that the temporal RSCDF extension must have are: *temporal marker*, *temporal relation* and *temporal trace*. The first assumes assigning to any resource state a temporal stamp; the second assumes defining a temporal relation between two resource conditions. The last one, temporal trace will be used as a container for sequences of resource states – the dynamics of their change through time.

2.1.2 Standardization efforts for temporal metadata

As most comprehensive standardization activity related to temporal metadata, DAML-Time effort [DAML-Time] can be mentioned. This is a collaborative project, which is led by Jerry Hobbs and is a part of the DARPA Agent Markup Language (DAML – [DAML]) project. The DAML-Time project "aims to develop a representative ontology of time that expresses temporal concepts and properties common to any formalization of time" [DAML-Time].

The DAML-Time effort originated from the significant needs of the Semantic Web community in the standardized temporal metadata, ontologies [Hobbs, 2002].

In the development of the RSCDF temporal extension we will orient ourselves at the results of the DAML-Time effort and most obviously the *rscdf:TempMark* concept will inherit from the *TemporalEntity* - a top-level concept in the DAML-Time ontology (see description of the latter and links to the documentation in [Pan & Hobbs, 2004]). This adoption is made by reason of RSCDF compatibility with widely-adopted standards and by reason of reuse of world-wide expertise in temporal metadata. As yet, the DAML-Time ontology has been developed to the quite vast extent, which makes it ready-to-use for real-life implementations (see [DAML-Time ontology]).

Among other comparably widely-adopted standards that contain aspects of temporal metadata is Dublin Core Metadata Initiative [DublinCore]. However, the temporal metadata elements that it defines (see DC:date, DC:created, DC:dateAccepted, DC:dateCopyrighted, DC:dateSubmitted, DC:Period, DC:Coverage elements in [DCMI Metadata Terms]) are too poor to be used in RSCDF. It is obvious, because the Dublin Core ontology is dedicated to other purposes than RSCDF, namely, for description of digital documents. Hence, Dublin Core cannot be used as a base for temporal extension of RSCDF.

One more effort related to the temporal metadata standardization and which is worth considering is TimeML – Markup Language for Temporal and Event Expressions [TimeML]. This project is funded by the ARDA organization and lead by James Pustejovsky. TimeML aims at developing "a robust specification language for events and temporal expressions in natural language" [TimeML]. Recently (April 2004) the project team issued a series of documentation supporting this standardization effort, particularly TimeML Specification (v1.1): see [TimeML docs] for details. Obviously, the initial orientation of TimeML to the natural language makes it inappropriate to utilization for the temporal RSCDF extension. The joint publication of the leaders – Jerry Hobbs and James Pustevjovsky - of the two mentioned projects (DAML-Time and TimeML respectively) (see [Hobbs & Pustejovsky, 2003]) makes an attempt of defining mappings between those two standards. The results of this cooperative analysis are still obscure and increase our initial opinion about TimeML as a standard not very applicable for the needs of the temporal RSDCF extension. Moreover, the current TimeML specification is fully based on XML standard [TimeML docs] and the absence of any representation that conforms to the Semantic Web based standards makes TimeML unready for its immediate practical utilization. In principle, most tasks (see in [Hobbs & Pustejovsky, 2003]), which TimeML basically addresses, are quite appropriate for the temporal RSCDF extension.

2.1.3 Implementation of the RSCDF temporal extension

In this subsection three requirements that were imposed on the temporal extension in Section 2.1.1 are fulfilled. They were namely *temporal marker*, *temporal relation* and *temporal trace*.

Temporal marker or temporal stamp assumes adding temporal features to resource state (*rscdf:State*) with help of a compound *rscdf:TempMark* ontological concept, which relates to a resource state through the *rscdf:inTime* property (see Figure 2).

In order to reuse the recent results in the domain of temporal metadata, the DAML-Time OWL ontology will be adopted as it was decided in Section 2.1.2. For the first version of RSCDF, we will orient at the sub-ontology of time, which was developed by Feng Pan and Jerry Hobbs [DAML-Time ontology]. It is also expressed in OWL and is much simpler than the full ontology and "provides all of what most users would need, i.e., a vocabulary for expressing facts about topological relations among instants, intervals, and events, together with information about durations and about dates and times" [DAML-Time].

Since the most upper class in that sub-ontology of time is *time-entry:TemporalThing*, *rscdf:TempMark* will relate directly to it through the property *rscdf:describedBy* (see Figure 2).

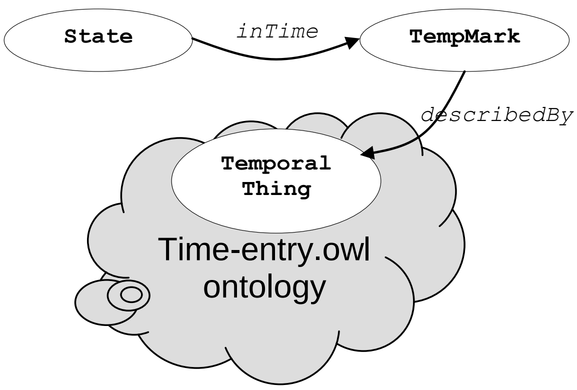

Figure 2 – Adoption of the *time-entry* sub-ontology

One may notice that the necessity in this potentially redundant *TempMark* concept in the RSCDF ontology is a disputable point. The property *inTime* could refer directly to *TemporalThing*; *TempMark* exists only to carry a specific semantic sense, defining the purposes of the *TemporalThing* utilization more exactly. Namely, the *TemporalThing* concept is used as a temporal marker or stamp put on a resource state.

As for the last requirement, namely a *temporal trace*, this kind of RDF extension is realized through the property *rscdf:hasHistory*, which belongs to *rscdf:SmartResource* and refers to a set of SmartResource's states that can increase permanently during a lifecycle of the resource (see Figure 3). There is an uncertainty concerning the parent class for that container: should it be a subclass of *rdf:Sequence* or *rdf:Bag*?

As it is well known, *Sequence* assumes ordering its members; hence, it must be used, if the history container will include a set of *States* ordered into a timeline. However, if different *TempMark* concepts associated with corresponding *State* concepts within the same history container could form temporal intersections, in this case the container would have to be declared as a descendant of *Bag*. Consider an example, related to the second type of the container declaration: one could describe *State* of *SmartResource* as a periodically occurring one. The latter can be, for instance, a systematic repair or inspecting of *SmartResource*, known in advance for a certain period in future. Ultimately, the mentioned container could be a descendant of the more general class *rdf:Container*, as it has been declared in the current version of RSCDF schema.

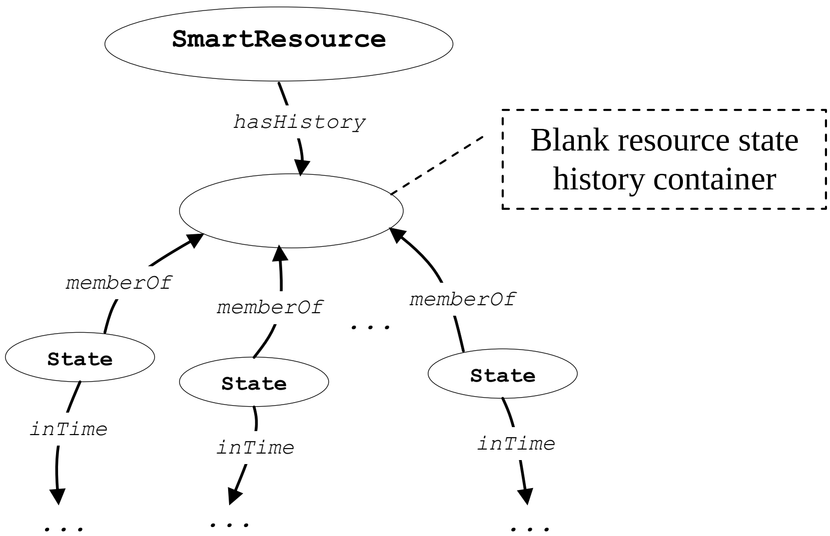

Figure 3 – Temporal trace realization for SmartResource

Now, having all important temporal features realized in RSCDF schema, it is clear, how to query vital data about a certain SmartResource. Most frequently performed requests for resource state can acquire values of a certain number of resource properties (spectrum) for a given point of time (a) and values of a given resource property (spectral constituent) through the time (b). To examine those cases, it is necessary to look a little deeper into the RSCDF schema. For this purpose it is necessary to know, that every resource *State* entry in the history has associated set of parameter value containers, which in their turn have specific related parameter type, denoted by the *ParamType* class.

Now, the case *a* can be performed via the following RQL-query [RQL], if to assume that a resource history will be stored on a Sesame RQL Server [openRDF] (see Figure 4).

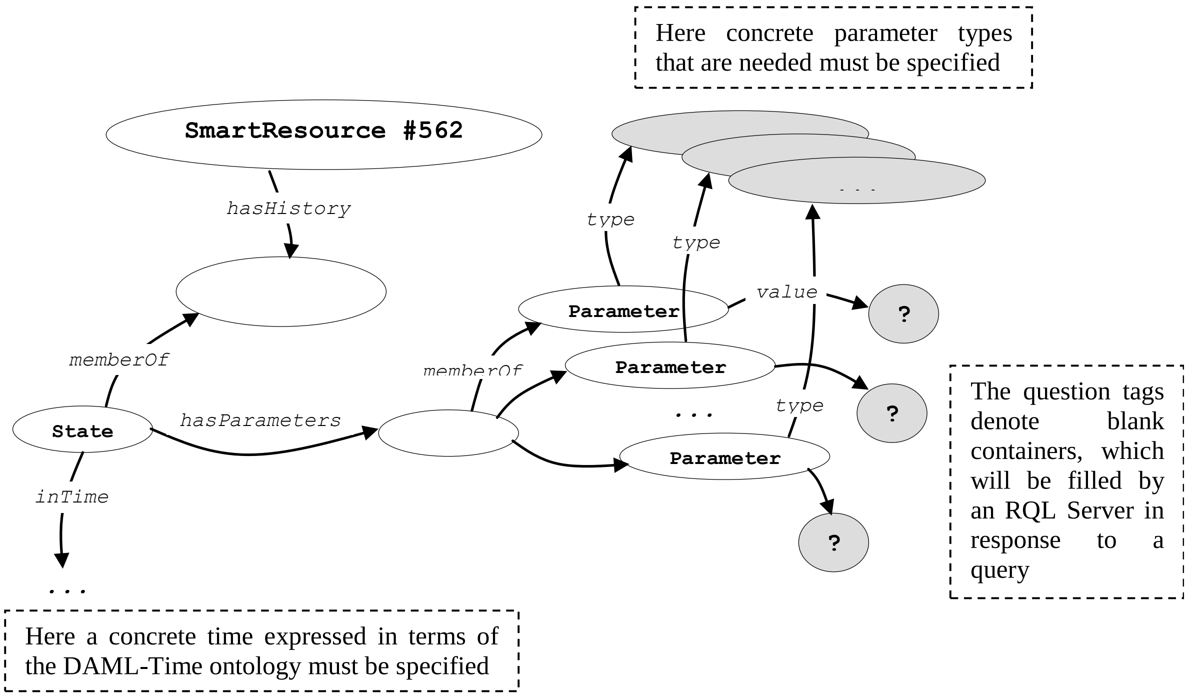

Figure 4 – Querying resource state history, case (a)

In the Figure 4 the necessary RQL-query is represented in a graphical form. It shows the structure of a graph that will be constructed in response to a query for values of a certain number of resource properties (spectrum) for a given point of time. The necessary parameter types specified by a user (in our case it will be a SmartResource) will be put in the containers filled by grey colour in the figure. Also, a user will have to specify a concrete time entity in the query, which must be expressed following the DAML-Time ontology. Sesame RQL Server, which will store a SmartResource history, will reply with corresponding parameter values enclosed in the blank containers instead of question marks.

The case *b* can be performed via the RQL-query, depicted in Figure 5, which is quite similar to the query used for the *a* case.

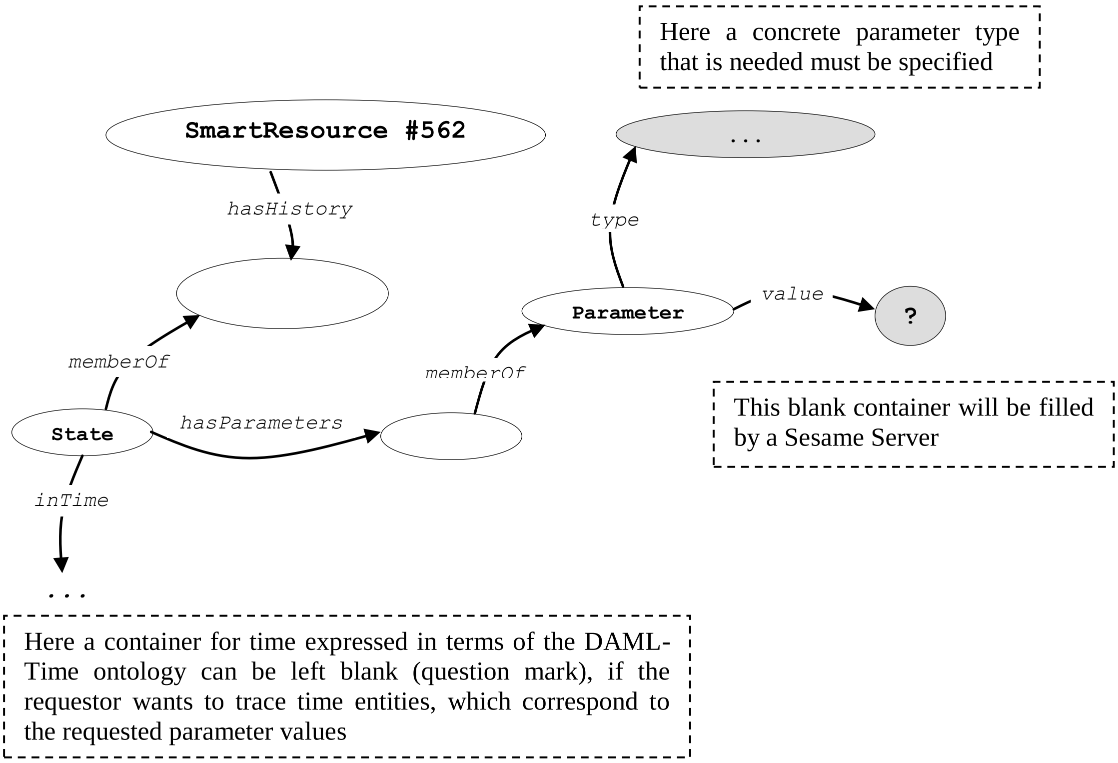

Figure 5 – Querying resource state history, case (b)

The requestor in order to fetch values of a given resource property (spectral constituent) through the time, must specify the type of that property/parameter. Optionally, the requesting side can specify a blank container to be filled by time entities, which correspond to the retrieved parameter values. Finally, the server reply will have a form of pairs: *time-entity – parameter-value*.

2.1.4 Further development of the RSCDF temporal extension

In next versions of the RSCDF temporal extension, Temporal Validity Intervals would be very useful, too. The latter are intended for expressing a time-to-live of a resource property and might be declared using *rdf:Statement* construction. This extension makes possible, for instance, temporal diagnoses (which expire after certain period of time or occur periodically) and opinions. In this case, a necessity in the specific maintenance agent, which changes the RDF-schema according to the time-to-live properties, must be analyzed carefully.

Time aggregation, which relates to a frequency of parameter value acquisition and precision in time, has to be realized within the RSCDF temporal extension, too. This aspect must be supported by Time Aggregation Ontology. Diagnostic Services must have a possibility to vary a time precision in a request for parameters values. Aggregation concepts can be used for realization of the archiving function for resource state that will be accumulated in the resource history ("Resource Lifeblog"). The archiving assumes generalization of the sequence of resource states by one term for the purpose of storage space saving and increasing access time to the resource properties. Some services for diagnostics may need just generalized description of the resource state during the given period of time instead of the detailed log of its precise states.

Current version of RSCDF still supports basic time aggregation. Let us consider an example of how to perform such aggregation using current possibilities of RSCDF. In this example we will focus just on one parameter that is included in the whole resource state among many other parameters-constituents. Assume that the parameter values change through the time as it is shown in Figure 6 below.

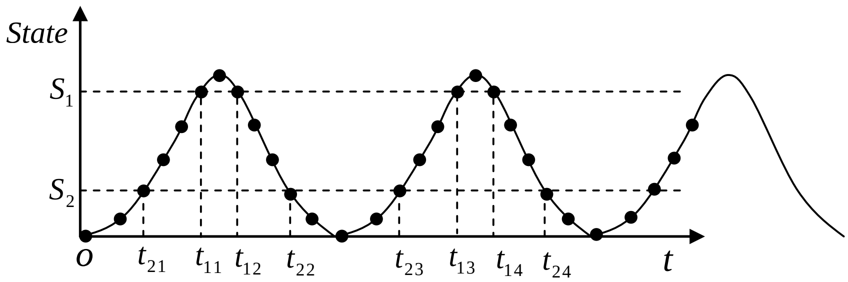

Figure 6 – Simple time aggregation in RSCDF

In Figure 6 the vertical axis denotes the resource states that occur throuh the time (the horizontal axis is dedicated to the time). In the Figure it is shown that the dynamics of the resource state change has a periodic character. Such cases are the most typical to be used for a simple time aggregation in RSCDF. In Figure 6, there are two instances of the resource state (parameter value) occurance: S_1 and S_2. One can see that, for example, state S_1 occures in the time moments $t_{11}, t_{12}, t_{13}, t_{14} ... t_{1n}$, which have a certain period. Hence, if the resource history contained a set of pairs $S_x; t_{xy}$, a long sequence of such pairs can be substituted by its aggregation in form of only one pair $S_x; t_{xagg}$. If to take the S_1 state as an example, its corresponding history sequence can be aggregated into the pair $S_1; t_{1agg}$, where S_1 instance remains the same, but t_{1agg} is a unit of two instant time entities and a time period that corresponds to them. The t_{1agg} concept must be expressed in the terms of the DAML-Time ontology.

The described simple time aggregation can be realized using the available means of the DAML-Time ontology. However, to support advanced time aggregation, like e.g. substitution of the state history sequence by its analytical representation (mathematical formula, etc.), specific concepts in RSCDF must be developed for this purpose.

2.2 Contextual RSCDF extension

[Sayers & Wilkinson, 2003] have discovered that with the original RDF it is quite complicated to represent and query collections of statements in context. They introduced an original higher-level object, the *Snippet*, to hold a fragment of RDF that is about a single subject and made within a particular context. Each snippet may be represented as a bag of reified statements where all the statements are about a single subject and all are made within a particular context. Approach allows also making statements about snippets and snippets about snippets when necessary.

[Guha *et. al., 2004*] suggested to construct a class of all data sources with their URLs in the ontology and to consider each instance of this class as a context for all data taken from appropriate data source. By that way it will be possible to extract the content, which is true-in-context of its data source. [Priebe, 2004] uses Dublin Core elements to describe the context in RDF. For that two Dublin Core attributes dc:subject and dc:coverage are considered as suitable subset for describing context. The concept of Named Graphs [Caroll *et. al.*, 2004] extends the syntax and semantics of RDF enabling multiple (named) RDF descriptions with different identities (e.g. contexts). A unique name and an RDF graph are assigned to each Named Graph, but Named Graph itself may have any other properties. Such extension is considered as possible foundation for the trust layer in Semantic Web. In context enabled RDF (cRDF), suggested by [Bizer, 2004], the ordinary RDF data model (subject-predicate-object) was expanded ad includes also the context identifier and stating identifier. The first one identifies context of a set of statings, the second one identifies a single stating. Based on cRDF, trust architecture has been presented, which allows the formulation of subjective and task-specific trust policies as a combination of reputation-, context- and content-based trust mechanisms. In [Klyne, 2001] context is considered as a container of reified RDF statements with additional properties associated with context. Statements may be associated with multiple contexts. An approach has been also suggested to allow statements about contextual containers to be applied automatically to its members. It was assumed that RDF mechanism based on statements and contexts allows complex systems descriptions without detailed ontology of these systems component set. [Sheth et al., 2004] consider contexts as captured from RDF descriptions areas of possible interests of different users. A user can define several ontological regions with different weights to specify the association types she is interested in. If the discovery process finds some associations passing through highly weighted contextual regions then they are considered relevant. More general attempts to contextualize ontologies include C-OWL (Context OWL), which has been proposed by [Bouquet *et al.*, 2004] as an extended language with an enriched semantics. It allows localizing contents of ontologies in different context (and, therefore, to make them not visible to the outside) and provide explicit mappings (bridge rules) for controlled global visibility. This is only the first step and a lot of research remains to be done. The core issue is the tension between how much we should share and globalize (via ontologies) and how much we should localize with limited and totally controlled forms of globalization (via contexts).

In earlier work [Terziyan & Puuronen, 1997] a Semantic Metanetwork was presented as modelling tool for knowledge, which is valid in certain context and allowing the context itself to be valid within some metacontext, etc. The context itself was considered in a broad meaning (source of information, time, space, etc.). The model includes powerful tools for reasoning with contexts, e.g. interpretation, decontextualization, lifting, discovering context, etc. Most of these properties, which are based on multilevel context, are quite complicated to realize within RDF.

Context Description Framework (CDF) was suggested as *lite* version of RSCDF in [Khriyenko & Terziyan, 2005] to stress contextual enhancement of the RDF by RSCDF. According to CDF vision, any property has some sense in certain context, which should be specified by the context tolerance range (see Figure 7). Each statement may be true or false concerning to different conditions of an environment. In this case a context of a statement is a set of other statements, which describe a certain condition (state) of an environment. Such descriptions among properties of an environment may contain also source of the statement descriptions, and thus provide opportunity to manage trust in distributed systems. Each contextual statement itself may also have an own context (i.e. nested context). We found out that using triplet-based model for a statement-in-context description is not suitable and we use quadruples (see Figure 8) for ontology modelling, where fourth additional component is a container of contextual statements.

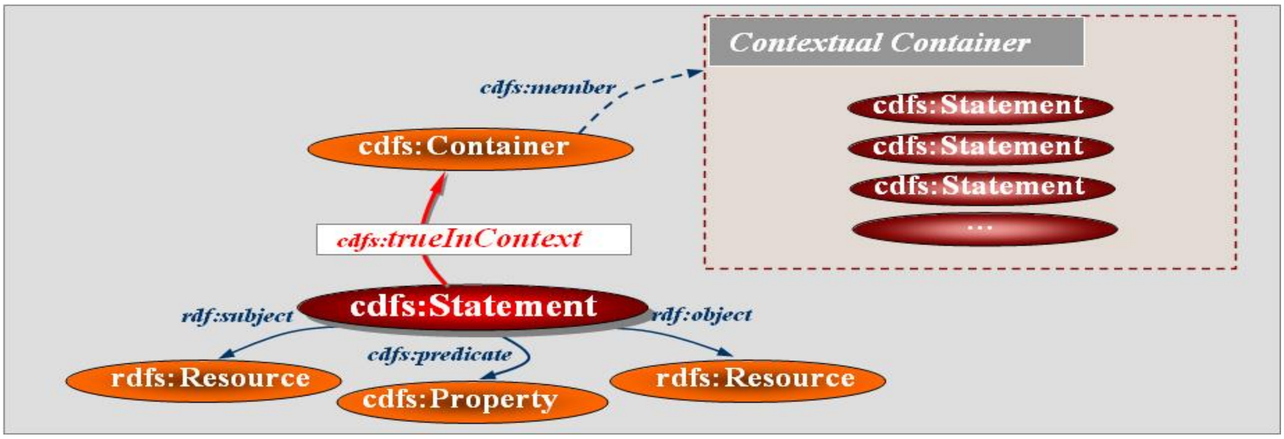

Figure 7 - A quadruple vision of the statement

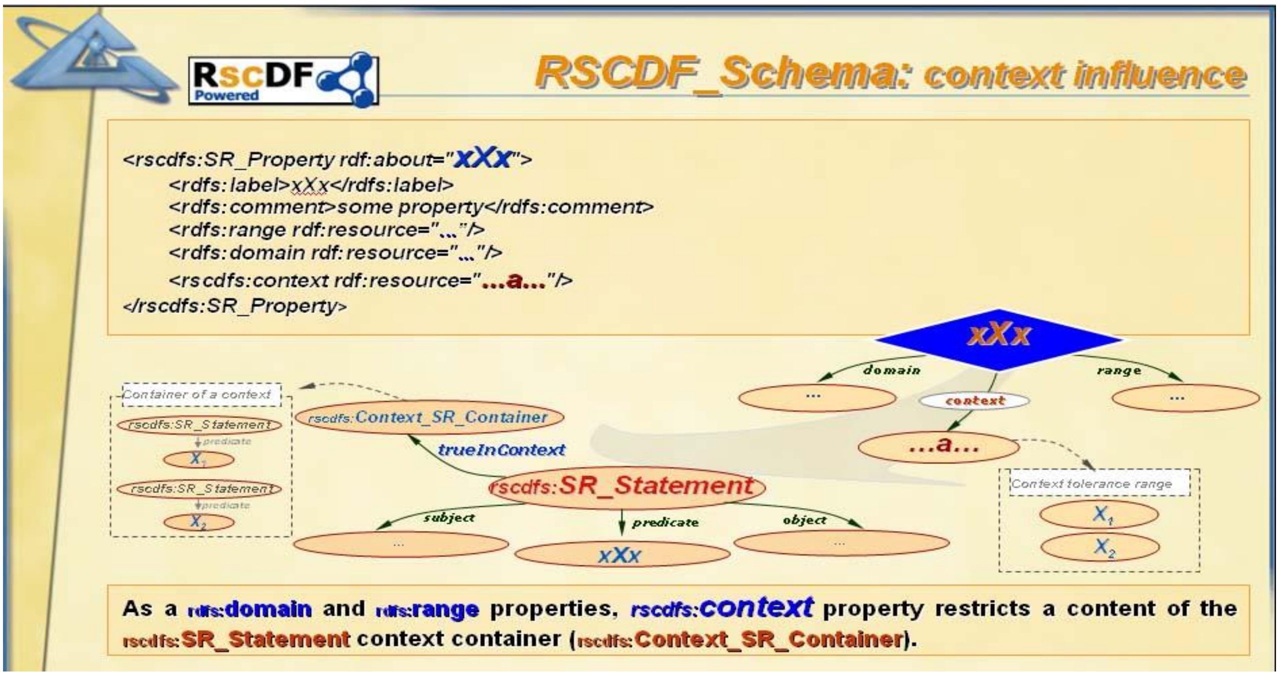

Figure 8 – Influence of a property contextual constraints on a RDF statement

Contextual RSCDF extension implies a possibility to assign additional properties to a series of resource properties (RDF subgraphs) – see Figure 9. This contextual extension can make possible assigning statements to a piece of a resource history (e.g. diagnosis of a doctor). This extension must provide a possibility of indicating a source of the statement (e.g. Web Service X135-F or Expert An-17).

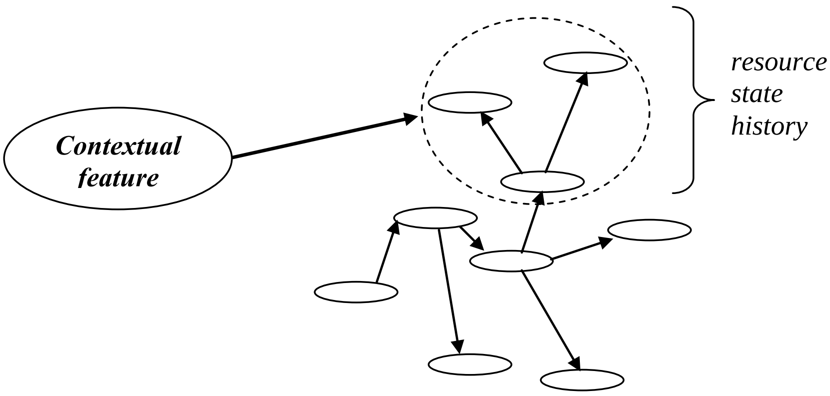

Figure 9 – Contextual RSCDF extension

3. Support for data types in RSCDF

RSCDF is a framework for adding metadata to the actual data. However, what is considered as data? Well, it can by standard data primitives, like Real, Integer, String. They will be enclosed into a necessary RDF-markup. Additionally, RSCDF in order to become universal must be provided with metadata concepts that will allow supporting compound data types, like XML, database formats, pieces of data formatted according a certain protocol. MathML can be used for aggregation of data about the dynamics of parameter values change. Thus, RDF must include upper-classes to denote different existing data formats.

While transforming legacy data format into the RSCDF-based, different levels of a structural detailed elaboration can be used. The initial data format can be totally transformed into the RSCDF-based format, broken down into data primitives. At the other extreme, initial data format can be wrapped by the RSCDF as it is with an indication of its format class. The RSCDF must provide this flexibility to make possible accelerated data exchange between active resources that are based on the same data format. The acceleration of the exchange is achieved through avoiding unnecessary data format transformations inside resource adapters. We also must take into account situations, when a partial interoperability between a pair of resources occurs on a data format level. When, for instance, both resources utilize Relational Databases as a base for data format, but use different structures of the databases.

It is also reasonable to provide for specifying an URI of the actual physical data storage, where the data reside, instead of inserting long data sequences into RDF-files. Is the URI specification developed enough to make such references to all existing data types (consider, for instance, a database, Flat Files)?

This aspect of the RSCDF opens a wide area for different Web Services dedicated for data transformation between different formats, different structural detailed elaboration levels. This aspect is also related to the challenge of *Service Orchestration*, which assumes composing sequences of Web Services, which have data in X format on its input and has data in Y format on its output.

Hence, corresponding ontology must include information about aggregation, normalization of values and method/service/type of a processing that were applied for the initial data. Most likely, these concepts will be included into Adaptation Framework, particularly on its Semantic Level (see Adaptation Layers). They might also enter into Behavioural Framework in form of instructions concerning data processing and transformation. In this context, a challenge of responsibilities distribution comes up: who will be responsible for a management of the data transformation process (search and orchestration of the necessary services): a Device or a Web Service?

The possible view of a part of ontology that reflects data format aspects is shown in Figure 10.

The core RSCDF triplets and the ones that belong to the RSCDF extension for DataFormat are separated by a dotted line. One can see that in the extension the SmartResource has a preliminary subclass SmartService, which corresponds to the abstract WebService component in the conceptual *Device-Expert-WebService* interaction triangle. SmartService, in its turn, has a subclass – DataFormatTransformator, which denotes a specific WebService, which performs transformation of one data format to another. Therefore, DataFormatTransformator has two properties: *input* and *output*, which relate it to the ParamType concept.

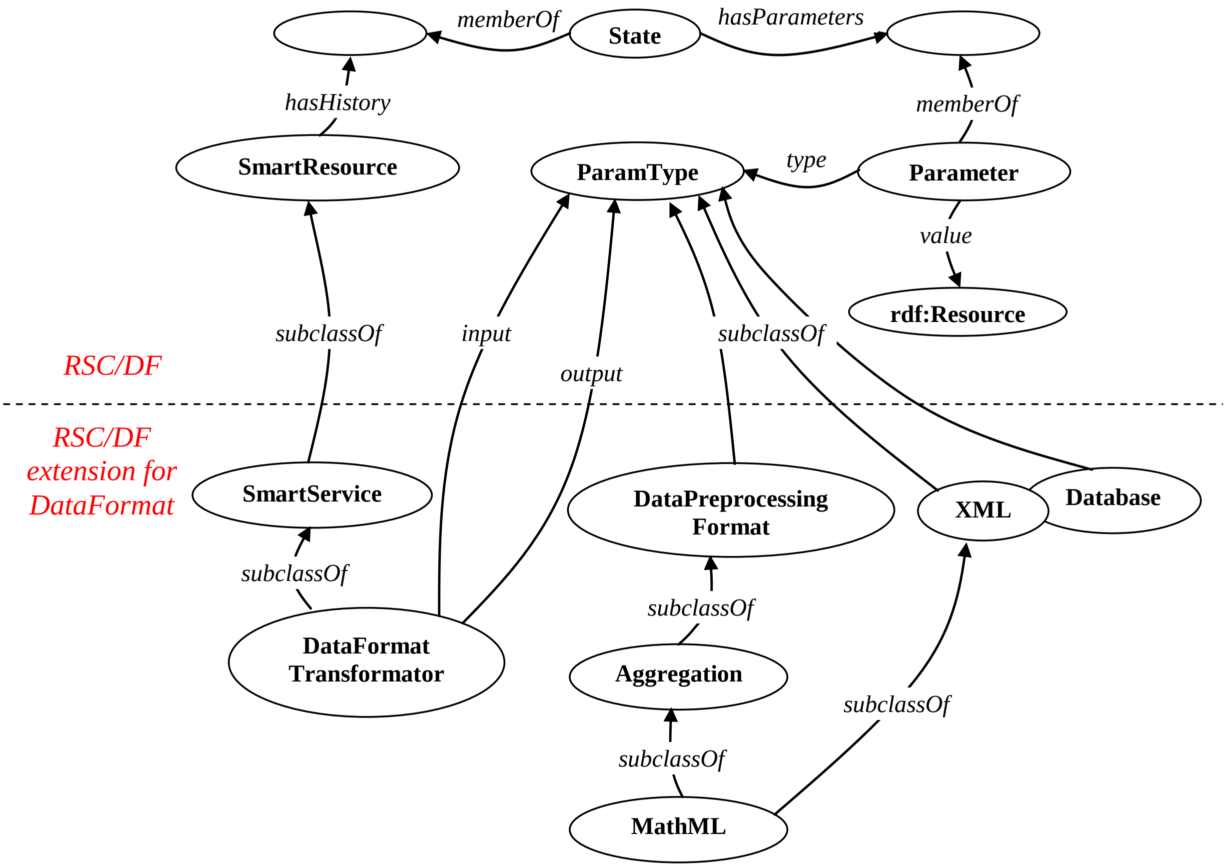

Figure 10 – RSCDF extension for data formats support

On the left of DataFormatTransformator, a preliminary ontology branch for different data format types is located. DataPreprocessingFormat serves as an upper class for all intermediate data formats. One can see an example extension of this upper class by the Aggregation class, which corresponds to aggregative data formats, discussed above. Aggregation has a child – MathML, which is a specific data format for aggregated real-time data representation.

It is assumed that descendants of DataFormatTranslator – concrete WebServices transformating data formats – will have their *input* and *output* properties, referred to more specific ParamTypes - its descendants from the mentioned ontology branch.

In the considered RSC/DF extension there are few subclasses of ParamType, which denote general data formats, like XML and Database data format. They are not intermediate data formats on their own; they are rather basic data formats. This part of ontology will be also branchy. The relations between this part of ontology and descendants of DataProcessingFormat are still obscure. As a preliminary, those relations are expected to be *subclassOf* (see this relation between MathML and XML).

4. SSDDM-structure for the RSCDF schema

SSDDM-model (State-Symptom-Diagnosis-Decision-Maintenance) must contain all necessary concepts, their properties and relations between the concepts to support the overall lifecycle of the Maintained Resource. The lifecycle includes the following stages:

1. Sensing resource state (a set of parameter values across a timeline) or reading it from the resource state history (resource state lifeblog). Essential is that the resource state can comprise

as its part opinions of the other resources about its state. The resource can grade these opinions by the parameter of their quality.

2. Making preliminary state processing (alarm systems), generating symptoms (preliminary assumption about a diagnosis). Alarm system is a Web Service, which implements an algorithm that makes a decision about a symptom based on the analysis of the resource state.

Below there are different cases of a location of Alarm Service in the general architecture of the maintenance system. Figure 11a depicts a case, when Alarm Service has been configured for a certain field device, for its specific format of state data. In this case the adapter is not involved into the data exchange process between a device and Alarm Service.

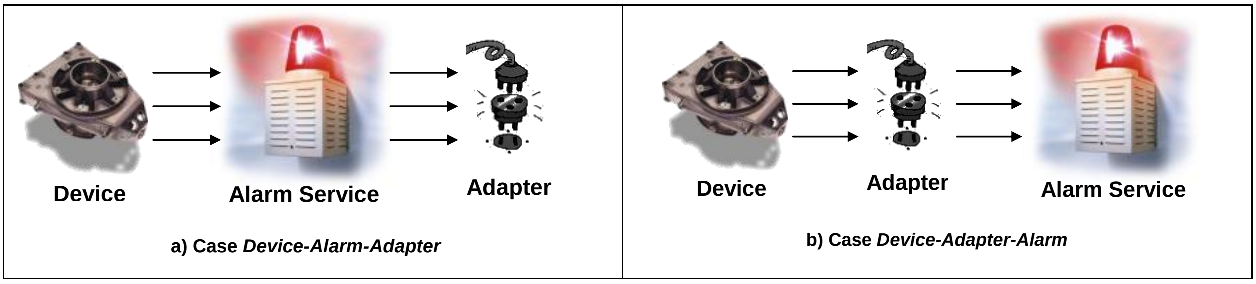

Figure 11 – Alarm Service – Adapter configurations

Figure 11b describes other case, when Alarm Service is external ("alien") with regard to Device. For this case, the state of the Device described in the internal standard, when transferred to the "alien" Alarm Service, come through Adapter for transformation into RSCDF standard.

3. Making detailed state processing (diagnostic services, previously generated symptoms can be taken into account) and making final decisions about diagnosis. Diagnosis can be complex (different diagnoses from different sources and distributed in time). Resource might store its "out-patient card", which will contain a history of diseases/diagnoses (resource diseases lifeblog). On this stage a notion of a "Doctor" can be distinguished as someone who sensing a resource state (mark) makes an advice about further actions in order to change the state (mark) to normal.

4. Performing necessary maintenance actions for a given resource based on the previously stated diagnoses. Thus, there must be a service that will generate a *recommendation* (a set of behavioral instructions) for a corresponding resource according to the assigned diagnosis. Maintenance actions must be put down to the resource history/lifeblog. System must "know" what has been made with it, a cause of changes happened with its parameters.

The SSDDM environment meant for online condition monitoring and predictive maintenance of various industrial resources. Utilization of RSCDF allows creation of agent-driven platforms for each industrial resource where all data related to monitoring, diagnostics and maintenance of the resource will be collected in the resource history ("lifeblog") and managed by the resource agent. The basic and more or less universal maintenance lifecycle of a resource (device, expert, service, etc.) and its contribution to the resource history is shown in Figure 12.

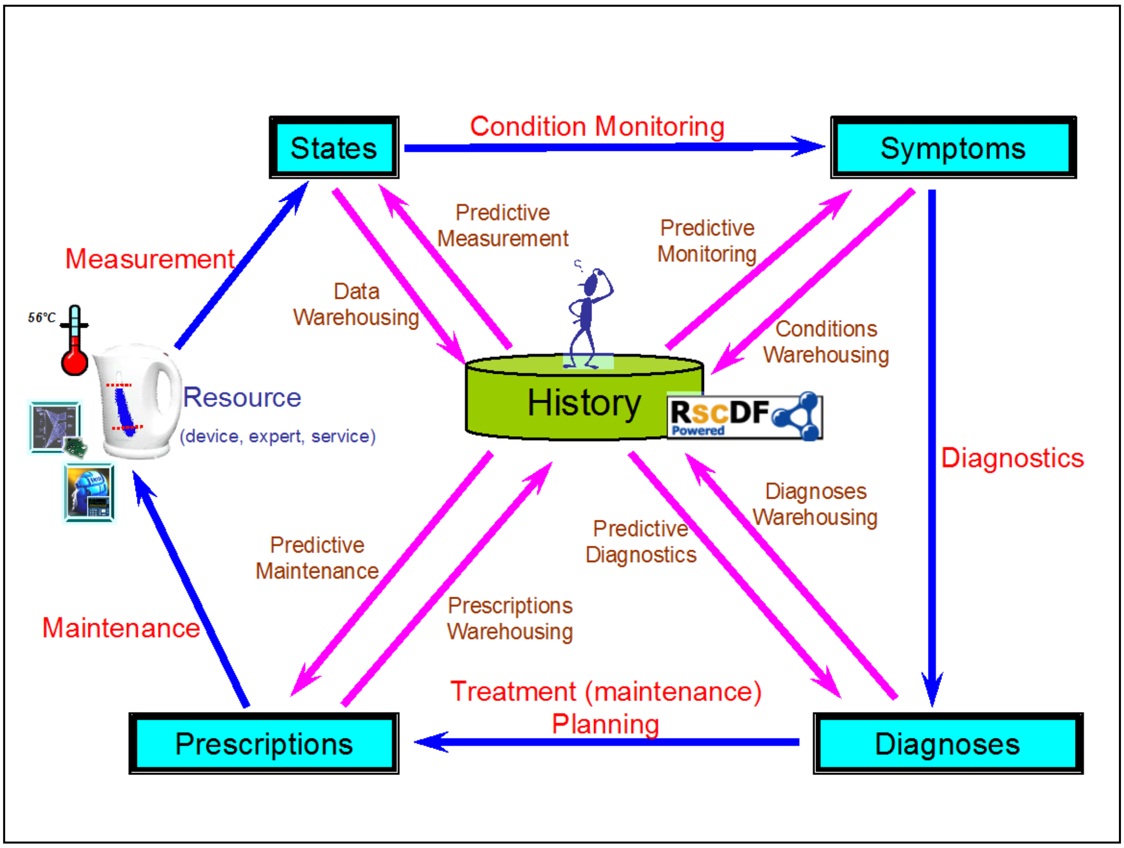

Figure 12 – SSDDM life cycle

5. RSCDF schema

The regular RSCDF elaboration process has resulted in the following ontology (*Hierarchy of Classes* and *Hierarchy of Properties*).

```
<?xml version='1.0' ?>

<!DOCTYPE rdf:RDF [
         <!ENTITY rdf 'http://www.w3.org/1999/02/22-rdf-syntax-ns#'>
         <!ENTITY rdfs 'http://www.w3.org/2000/01/rdf-schema#'>
         <!ENTITY rscdfs
'http://www.cc.jyu.fi/~olkhriye/rscdfs/0.3/rscdfs#'>]>

<rdf:RDF
    xmlns:rdf="&rdf;"
    xmlns:rdfs="&rdfs;"
    xmlns:rscdfs="&rscdfs;" >
```

- *SmartResource* concept generalizes all SmartResources that will be integrated into the target system of self-maintained resources.

```
<rscdfs:SmartResource rdf:about="&rscdfs;SmartResource">
  <rdfs:comment>Type of itself</rdfs:comment>
  <rdfs:label>SmartResource</rdfs:label>
  <rdfs:subClassOf rdf:resource="&rdfs;Class"/>
</rscdfs:SmartResource>
```

- `rscdfs:Device` is a upper-class for all devices in the target system of self-maintained resources.

```
<rscdfs:Device rdf:about="&rscdfs;Device">
  <rdfs:comment>Type of itself</rdfs:comment>
  <rdfs:label>Device</rdfs:label>
  <rdfs:subClassOf rdf:resource="&rscdfs;SmartResource"/>
</rscdfs:Device>
```

- RSCDF defines a specific branch of hierarchy that extends the SmartResource concept. It is a branch of decision makers (DecideSR): Services (machine learning algorithms) and Experts (human).

```
<rscdfs:Environment rdf:about="&rscdfs;DecideSR">
  <rdfs:comment>Type of itself</rdfs:comment>
  <rdfs:label>DecideSR</rdfs:label>
  <rdfs:subClassOf rdf:resource="&rscdfs;SmartResource"/>
</rscdfs:Environment>
<rscdfs:Service rdf:about="&rscdfs;Service">
  <rdfs:comment>Type of itself</rdfs:comment>
  <rdfs:label>Service</rdfs:label>
  <rdfs:subClassOf rdf:resource="&rscdfs;DecideSR"/>
</rscdfs:Service>
<rscdfs:Expert rdf:about="&rscdfs;Expert">
  <rdfs:comment>Type of itself</rdfs:comment>
  <rdfs:label>Expert</rdfs:label>
  <rdfs:subClassOf rdf:resource="&rscdfs;DecideSR"/>
</rscdfs:Expert>
```

- *SmartMessage* concept generalizes all message packages as proactive and self-maintained SmartResources.

```
<rscdfs:SmartMessage rdf:about="&rscdfs;SmartMessage">
  <rdfs:comment>Type of itself</rdfs:comment>
  <rdfs:label>SmartMessage</rdfs:label>
  <rdfs:subClassOf rdf:resource="&rscdfs;SmartResource"/>
</rscdfs:SmartMessage>
```

- *ResourceAgent* concept represents an agent of a SmartResource, which also may be considered as a SmartResource of a system as well as its resource.

```
<rscdfs:ResourceAgent rdf:about="&rscdfs;ResourceAgent">
  <rdfs:comment>Type of itself</rdfs:comment>
  <rdfs:label>ResourceAgent</rdfs:label>
  <rdfs:subClassOf rdf:resource="&rscdfs;SmartResource"/>
</rscdfs:ResourceAgent>
```

- *Environment* concept also represents a specific SmartResourece of the system. It is the one of the significant object for the context description.

```
<rscdfs:Environment rdf:about="&rscdfs;Environment">
  <rdfs:comment>Type of itself</rdfs:comment>
  <rdfs:label>Environment</rdfs:label>
  <rdfs:subClassOf rdf:resource="&rscdfs;SmartResource"/>
</rscdfs:Environment>
```

- *SmartResource* concept describes a machine learning model of a decision maker.

```
<rscdfs:SR_Model rdf:about="&rscdfs;SR_Model">
  <rdfs:comment>Type of itself</rdfs:comment>
  <rdfs:label>SmartResource Model</rdfs:label>
  <rdfs:subClassOf rdf:resource="&rdfs;Class"/>
</rscdfs:SR_Model>
```

- This class is reserved with a purpose to describe the device's parameters.

```
<rscdfs:DParamDescription rdf:about="&rscdfs;DParamDescription">
  <rdfs:comment>Type of itself</rdfs:comment>
  <rdfs:label>Device parameter description</rdfs:label>
  <rdfs:subClassOf rdf:resource="&rdfs;Class"/>
</rscdfs:DParamDescription>
```

- `SR_Container` *(subclass of rdfs:Container)* is a upper-class for all containers in RSCDF.

```
<rdfs:Class rdf:about="&rscdfs;SR_Container">
  <rdfs:comment></rdfs:comment>
  <rdfs:label>SR_Container</rdfs:label>
  <rdfs:subClassOf rdf:resource="&rdfs;Container"/>
</rdfs:Class>
```

- *Contextual container for a RSCDF statement*. It contains just contextual statements with the correspondent statement predicates.

```
<rdfs:Class rdf:about="&rscdfs;Context_SR_Container">
  <rdfs:comment></rdfs:comment>
  <rdfs:label>Context_SR_Container</rdfs:label>
  <rdfs:subClassOf rdf:resource="&rscdfs;SR_Container"/>
</rdfs:Class>
```

- `rscdfs:SR_Statement` *(subclass of the rdf:Statement)* define a RSCDF statement concept with a correspondence to the vision (mentioned before) of quadruple representation of the statements.

```
<rdfs:Class rdf:about="&rscdfs;SR_Statement">
  <rdfs:comment></rdfs:comment>
  <rdfs:label>SR_Statement</rdfs:label>
  <rdfs:subClassOf rdf:resource="&rdf;Statement"/>
</rdfs:Class>
```

- `rscdfs:SR_Property` *(subclass of the rdf:Property)* define a RSCDF property concept accordingly to the triple representation of the property (contextual influence of the properties on each other) which was mentioned in the previous chapter.

```
<rscdfs:SR_Property rdf:about="&rscdfs;SR_Property">
  <rdfs:comment>Type of itself</rdfs:comment>
  <rdfs:label>SR_Property</rdfs:label>
  <rdfs:subClassOf rdf:resource="&rdfs;Class"/>
  <rdfs:subClassOf rdf:resource="&rdf;Property"/>
</rscdfs:SR_Property>
```

- All classes for definition of the quantities values are presented by upper-class `rscdfs:QuantityValue`. It is a branch point for the `rscdfs:NumericalValue` and `rscdfs:EnumerativeValue` subclasses (which are represents correspondent values descriptions).

```
<rscdfs:QuantityValue rdf:about="&rscdfs;QuantityValue">
  <rdfs:comment>Type of itself</rdfs:comment>
  <rdfs:label>QuantityValue</rdfs:label>
  <rdfs:subClassOf rdf:resource="&rdfs;Class"/>
</rscdfs:QuantityValue>
<rscdfs:NumericalValue rdf:about="&rscdfs;NumericalValue">
  <rdfs:comment>Type of itself</rdfs:comment>
  <rdfs:label>NumericalValue</rdfs:label>
  <rdfs:subClassOf rdf:resource="&rscdfs;QuantityValue"/>
</rscdfs:NumericalValue>
<rscdfs:EnumerativeValue rdf:about="&rscdfs;EnumerativeValue">
  <rdfs:comment>Type of itself</rdfs:comment>
  <rdfs:label>EnumerativeValue</rdfs:label>
  <rdfs:subClassOf rdf:resource="&rscdfs;QuantityValue"/>
</rscdfs:EnumerativeValue>
```

- `rscdfs:TempMark` *as a subclass of the* `rscdfs:EnumerativeValue` *plays a role of upper-class for all instances which are represent a time description.*

```
<rscdfs:TempMark rdf:about="&rscdfs;TempMark">
  <rdfs:comment>Type of itself</rdfs:comment>
  <rdfs:label>TempMark</rdfs:label>
  <rdfs:subClassOf rdf:resource="&rscdfs;EnumerativeValue"/>
</rscdfs:TempMark>
```

- Class for definition of the measurements units.

```
<rscdfs:MeasurementUnit rdf:about="&rscdfs;MeasurementUnit">
  <rdfs:comment>Type of itself</rdfs:comment>
  <rdfs:label>MeasurementUnit</rdfs:label>
  <rdfs:subClassOf rdf:resource="&rdfs;Class"/>
</rscdfs:MeasurementUnit>
```

- One of the significant extensions of the RDF within the vision of the triple representation of the property – third main property of the RSCDF-properties (domain, range, ***context***).

```
<rdf:Property rdf:about="&rscdfs;context">
<rdfs:isDefinedBy
rdf:resource="http://www.cc.jyu.fi/~olkhriye/rscdfs/0.3/rscdfs#"/>
 <rdfs:label>context</rdfs:label>
 <rdfs:comment>A context of the subject property.</rdfs:comment>
 <rdfs:range rdf:resource="&rscdfs;SR_Property"/>
 <rdfs:domain rdf:resource="&rscdfs;SR_Property"/>
</rdf:Property>
```

- Another one significant extension of the RDF within the vision of the quadruple representation of the statements – forth main property of the RSCDF-statements (subject, predicate, object, ***trueInContext/falseInContext***).

```
<rdf:Property rdf:about="&rscdfs;trueInContext">
  <rdfs:label>trueInContext</rdfs:label>
  <rdfs:domain rdf:resource="&rscdfs;SR_Statement"/>
  <rdfs:range rdf:resource="&rscdfs;Context_SR_Container"/>
</rdf:Property>
<rdf:Property rdf:about="&rscdfs;falseInContext">
  <rdfs:label>falseInContext</rdfs:label>
  <rdfs:domain rdf:resource="&rscdfs;SR_Statement"/>
  <rdfs:range rdf:resource="&rscdfs;Context_SR_Container"/>
</rdf:Property>
```

- The set of properties for the statements which define maintenance of a SmartResource, local time and time of the system (common environment time), measurement of SmartResource in certain time accordingly.

```
<rscdfs:SR_Property rdf:about="&rscdfs;maintenance">
  <rdfs:label>maintenance</rdfs:label>
  <rdfs:domain rdf:resource="&rscdfs;SmartResource"/>
  <rdfs:range rdf:resource="&rscdfs;QuantityValue"/>
</rscdfs:SR_Property>
<rscdfs:SR_Property rdf:about="&rscdfs;time">
  <rdfs:label>time</rdfs:label>
  <rdfs:domain rdf:resource="&rscdfs;SmartResource"/>
  <rdfs:range rdf:resource="&rscdfs;TempMark"/>
</rscdfs:SR_Property>
<rscdfs:SR_Property rdf:about="&rscdfs;sysTime">
  <rdfs:label>sysTime</rdfs:label>
  <rdfs:domain rdf:resource="&rscdfs;Environment"/>
  <rdfs:range rdf:resource="&rscdfs;TempMark"/>
  <rdfs:subPropertyOf rdf:resource="&rscdfs;time"/>
</rscdfs:SR_Property>
<rscdfs:SR_Property rdf:about="&rscdfs;measurement">
  <rdfs:label>measurement</rdfs:label>
  <rdfs:domain rdf:resource="&rscdfs;SmartResource"/>
  <rdfs:range rdf:resource="&rscdfs;QuantityValue"/>
  <rscdfs:context rdf:resource="&rscdfs;sysTime "/>
</rscdfs:SR_Property>
```

- RSCDF defines a specific branch of hierarchy that extends the Condition concept: Diagnosis and Alarm. Condition is a statement of one SmartResource about the state of other SmartResource derived based on a number of States. That is why among contextual statements for the condition (decision maker, time, decision model, etc.) at least one statement is a statement about SmartResource states sub history (rscdfs:sr_StateHistory). Diagnosis uses the `rscdfs:alarm` property to refer to a alarm situation (symptom), which are used for deriving the decision about the diagnosis and documentation purpose.

```
<rscdfs:SR_Property rdf:about="&rscdfs;condition">
  <rdfs:label>condition</rdfs:label>
  <rdfs:domain rdf:resource="&rscdfs;SmartResource"/>
  <rdfs:range rdf:resource="&rscdfs;QuantityValue"/>
  <rscdfs:context rdf:resource="&rscdfs;sysTime"/>
  <rscdfs:context rdf:resource="&rscdfs;decideSR_PartOf"/>
  <rscdfs:context rdf:resource="&rscdfs;model"/>
```

```
  <rscdfs:context rdf:resource="&rscdfs;sr_StateHistory"/>
</rscdfs:SR_Property>
<rscdfs:SR_Property rdf:about="&rscdfs;diagnosis">
  <rdfs:label>diagnosis</rdfs:label>
  <rdfs:domain rdf:resource="&rscdfs;SmartResource"/>
  <rdfs:range rdf:resource="&rscdfs;QuantityValue"/>
  <rscdfs:context rdf:resource="&rscdfs;sysTime"/>
  <rscdfs:context rdf:resource="&rscdfs;decideSR_PartOf"/>
  <rscdfs:context rdf:resource="&rscdfs;model"/>
  <rscdfs:context rdf:resource="&rscdfs;alarm"/>
  <rscdfs:context rdf:resource="&rscdfs;sr_StateHistory"/>
  <rdfs:subPropertyOf rdf:resource="&rscdfs;condition"/>
</rscdfs:SR_Property>
<rscdfs:SR_Property rdf:about="&rscdfs;alarm">
  <rdfs:label>alarm</rdfs:label>
  <rdfs:domain rdf:resource="&rscdfs;SmartResource"/>
  <rdfs:range rdf:resource="&rscdfs;QuantityValue"/>
  <rscdfs:context rdf:resource="&rscdfs;sysTime"/>
  <rscdfs:context rdf:resource="&rscdfs;decideSR_PartOf"/>
  <rscdfs:context rdf:resource="&rscdfs;model"/>
  <rscdfs:context rdf:resource="&rscdfs;sr_StateHistory"/>
  <rdfs:subPropertyOf rdf:resource="&rscdfs;condition"/>
</rscdfs:SR_Property>
```

- `rscdfs:transformation` plays a role of the predicate for a statement which states transformation of the statement's subject to the statement's object in the context of the decision maker and the model.

```
<rscdfs:SR_Property rdf:about="&rscdfs;transformation">
  <rdfs:label>transformation</rdfs:label>
  <rdfs:domain rdf:resource="&rscdfs;SR_Statement"/>
  <rdfs:range rdf:resource="&rscdfs;SR_Statement"/>
  <rscdfs:context rdf:resource="&rscdfs;decideSR_PartOf"/>
  <rscdfs:context rdf:resource="&rscdfs;model"/>
</rscdfs:SR_Property>
```

- `rscdfs:model` plays a role of the predicate for a statement which defines knowledge model in the context of learning set (sub history of the labeled conditions).

```
<rscdfs:SR_Property rdf:about="&rscdfs;model">
  <rdfs:label>model</rdfs:label>
  <rdfs:domain rdf:resource="&rscdfs;SmartResource"/>
  <rdfs:range rdf:resource="&rscdfs;SR_Model"/>
  <rscdfs:context rdf:resource="&rscdfs;sysTime"/>
  <rscdfs:context rdf:resource="&rscdfs;sr_ConditionHistory"/>
</rscdfs:SR_Property>
```

- The general property that plays a role of the predicate for a statement which states that certain SmartResource has a container (some set of the features).

```
<rscdfs:SR_Property rdf:about="&rscdfs;has_Container">
  <rdfs:label>has_Container</rdfs:label>
  <rdfs:domain rdf:resource="&rscdfs;SmartResource"/>
  <rdfs:range rdf:resource="&rscdfs;SR_Container"/>
</rscdfs:SR_Property>
```

- State serves as a container for parameters of SmartResources along with their values measured by a sensor during a certain period of time. Theoretically, it relies on an assumption that resource's state is defined as a set of all its parameters with concrete values (measurements) during a given period of time. That is why `rscdfs:measurement` property plays a role of contextual filter for the target container.

```
<rscdfs:SR_Property rdf:about="&rscdfs;sr_State">
  <rdfs:label>sr_State</rdfs:label>
  <rdfs:domain rdf:resource="&rscdfs;SmartResource"/>
  <rdfs:range rdf:resource="&rscdfs;SR_Container"/>
  <rscdfs:context rdf:resource="&rscdfs;measurement"/>
  <rdfs:subPropertyOf rdf:resource="&rscdfs;has_Container"/>
</rscdfs:SR_Property>
```

- There is a set of the properties which aims to define sub histories for States and Conditions (Diagnosis and Alarms). The properties: `rscdfs:sr_State`, `rscdfs:condition`, `rscdfs:diagnosis` and `rscdfs:alarm` play roles of contextual filters for the target containers correspondently.

```
<rscdfs:SR_Property rdf:about="&rscdfs;sr_StateHistory">
  <rdfs:label>sr_StateHistory</rdfs:label>
  <rdfs:domain rdf:resource="&rscdfs;SmartResource"/>
  <rdfs:range rdf:resource="&rscdfs;SR_Container"/>
  <rscdfs:context rdf:resource="&rscdfs;sr_State"/>
  <rdfs:subPropertyOf rdf:resource="&rscdfs;has_Container"/>
</rscdfs:SR_Property>
<rscdfs:SR_Property rdf:about="&rscdfs;sr_ConditionHistory">
  <rdfs:label>sr_ConditionHistory</rdfs:label>
  <rdfs:domain rdf:resource="&rscdfs;SmartResource"/>
  <rdfs:range rdf:resource="&rscdfs;SR_Container"/>
  <rscdfs:context rdf:resource="&rscdfs;condition"/>
  <rdfs:subPropertyOf rdf:resource="&rscdfs;has_Container"/>
</rscdfs:SR_Property>
<rscdfs:SR_Property rdf:about="&rscdfs;sr_DiagnosisHistory">
  <rdfs:label>sr_DiagnosisHistory</rdfs:label>
  <rdfs:domain rdf:resource="&rscdfs;SmartResource"/>
  <rdfs:range rdf:resource="&rscdfs;SR_Container"/>
  <rscdfs:context rdf:resource="&rscdfs;diagnosis"/>
  <rdfs:subPropertyOf rdf:resource="&rscdfs;sr_ConditionHistory"/>
</rscdfs:SR_Property>
<rscdfs:SR_Property rdf:about="&rscdfs;sr_AlarmHistory">
  <rdfs:label>sr_AlarmHistory</rdfs:label>
  <rdfs:domain rdf:resource="&rscdfs;SmartResource"/>
  <rdfs:range rdf:resource="&rscdfs;SR_Container"/>
  <rscdfs:context rdf:resource="&rscdfs;alarm"/>
  <rdfs:subPropertyOf rdf:resource="&rscdfs;sr_ConditionHistory"/>
</rscdfs:SR_Property>
```

- *PartOf* concept generalizes all part_of relationships between the SmartResources.

```
<rscdfs:SR_Property rdf:about="&rscdfs;partOf">
  <rdfs:label>partOf</rdfs:label>
  <rdfs:domain rdf:resource="&rscdfs;SmartResource"/>
  <rdfs:range rdf:resource="&rscdfs;SmartResource"/>
</rscdfs:SR_Property>
<rscdfs:SR_Property rdf:about="&rscdfs;decideSR_PartOf">
  <rdfs:label>decideSR_PartOf</rdfs:label>
  <rdfs:domain rdf:resource="&rscdfs;DecideSR"/>
  <rdfs:range rdf:resource="&rscdfs;SmartResource"/>
  <rdfs:subPropertyOf rdf:resource="&rscdfs;partOf"/>
</rscdfs:SR_Property>
<rscdfs:SR_Property rdf:about="&rscdfs;device_PartOf">
  <rdfs:label>device_PartOf</rdfs:label>
  <rdfs:domain rdf:resource="&rscdfs;Device"/>
  <rdfs:range rdf:resource="&rscdfs;SmartResource"/>
  <rdfs:subPropertyOf rdf:resource="&rscdfs;partOf"/>
</rscdfs:SR_Property>
<rscdfs:SR_Property rdf:about="&rscdfs;service_PartOf">
  <rdfs:label>service_PartOf</rdfs:label>
  <rdfs:domain rdf:resource="&rscdfs;Service"/>
  <rdfs:range rdf:resource="&rscdfs;SmartResource"/>
  <rdfs:subPropertyOf rdf:resource="&rscdfs;decideSR_PartOf"/>
</rscdfs:SR_Property>
```

- *PartOfWorld* concept aims to describe a SmartResource as an element of the first level of the part_of hierarchy.

```
<rscdfs:SR_Property rdf:about="&rscdfs;partOfWorld">
  <rdfs:label>partOfWorld</rdfs:label>
  <rdfs:domain rdf:resource="&rscdfs;SmartResource"/>
  <rdfs:range rdf:resource="&rscdfs;Environment"/>
  <rdfs:subPropertyOf rdf:resource="&rscdfs;partOf"/>
</rscdfs:SR_Property>
<rscdfs:SR_Property rdf:about="&rscdfs;decideSR_partOfWorld">
```

```
  <rdfs:label>partOfWorld</rdfs:label>
  <rdfs:domain rdf:resource="&rscdfs;DecideSR"/>
  <rdfs:range rdf:resource="&rscdfs;Environment"/>
  <rdfs:subPropertyOf rdf:resource="&rscdfs;decideSR_PartOf"/>
</rscdfs:SR_Property>
```

- *AtomOf* concept aims to describe a SmartResource as a leaf - an element of the last level of the part_of hierarchy.

```
<rscdfs:SR_Property rdf:about="&rscdfs;atomOf">
  <rdfs:label>atomOf</rdfs:label>
  <rdfs:domain rdf:resource="&rscdfs;SmartResource"/>
  <rdfs:range rdf:resource="&rscdfs;SmartResource"/>
  <rdfs:subPropertyOf rdf:resource="&rscdfs;partOf"/>
</rscdfs:SR_Property>
```

- Redefined properties in the context of new SR_Statement and SR_Container concepts.

```
<rdf:Property rdf:about="&rscdfs;predicate">
  <rdfs:label>predicate</rdfs:label>
  <rdfs:domain rdf:resource="&rscdfs;SR_Statement"/>
  <rdfs:range rdf:resource="&rscdfs;SR_Property"/>
  <rdfs:subPropertyOf rdf:resource="&rdf;predicate"/>
</rdf:Property>
<rdf:Property rdf:about="&rscdfs;member">
  <rdfs:label>member</rdfs:label>
  <rdfs:domain rdf:resource="&rscdfs;SR_Container"/>
  <rdfs:range rdf:resource="&rscdfs;SR_Statement"/>
  <rdfs:subPropertyOf rdf:resource="&rdfs;member"/>
</rdf:Property>
```

- *Properties for the values and unit description*.

```
<rdf:Property rdf:about="&rscdfs;value">
  <rdfs:label>value</rdfs:label>
  <rdfs:domain rdf:resource="&rscdfs;NumericalValue"/>
  <rdfs:range rdf:resource="&rdfs;Literal"/>
  <rdfs:subPropertyOf rdf:resource="&rdf;value"/>
</rdf:Property>
<rdf:Property rdf:about="&rscdfs;minValue">
  <rdfs:label>minValue</rdfs:label>
  <rdfs:domain rdf:resource="&rscdfs;DParamDescroption"/>
  <rdfs:range rdf:resource="&rdfs;Literal"/>
</rdf:Property>
<rdf:Property rdf:about="&rscdfs;maxValue">
  <rdfs:label>maxValue</rdfs:label>
  <rdfs:domain rdf:resource="&rscdfs;DParamDescroption"/>
  <rdfs:range rdf:resource="&rdfs;Literal"/>
</rdf:Property>
<rdf:Property rdf:about="&rscdfs;unit">
  <rdfs:label>unit</rdfs:label>
  <rdfs:domain rdf:resource="&rscdfs;NumericalValue"/>
  <rdfs:range rdf:resource="&rscdfs;MeasurementUnit"/>
</rdf:Property>
```

- The property for the description of the static SmartResource’s (device’s) parameters.

```
<rdf:Property rdf:about="&rscdfs;hasParameter">
  <rdfs:label>hasParameter</rdfs:label>
  <rdfs:domain rdf:resource="&rscdfs;Device"/>
  <rdfs:range rdf:resource="&rscdfs;DParamDescroption"/>
</rdf:Property>

</rdf:RDF>
```

A concrete example of the use of RSCDF for describing history of an industrial device states and conditions is given in Appendix.

6. Conclusions

In this paper we have presented RSCDF as temporal and contextual extensions of RDF and also presented an RSCDF-Schema based on State-Symptom-Diagnosis-Decision-Maintenance model. We believe that RSCDF will be a useful extension and allow describing state and condition of any Web-connected industrial resource in a machine-processable form. Collected RSCDF history of any industrial device states and conditions would be an appropriate sample (training) set for a machine-learning algorithm (as Web-Service) to create a model of the device and to enable further automatic diagnostics and prediction of it for predictive maintenance needs.

During the elaboration of the RSCDF schema a couple of controversial points have emerged. Are the concepts that denote input and output informational interfaces of the SmartResource necessary in addition to the concepts that denote internal resource state? Input-output concepts in the ontology will be useful for tasks of Information Flows Analysis inside the Maintenance Environment. During this analysis every node in the system (SmartResource) is considered as a “black box”.

Also, is it properly to declare and elaborate subclasses of the SmartResource: a Human (Industrial Expert), a Service (Industrial Web Service, Maintenance Center), a Device, etc.? They might be considered as domain-specific extensions, but on the other hand they are fair for many other domains: quite many domains contain these three main players within some or other interpretation.

Acknowledgements

We are grateful to Tekes (National Technology Agency of Finland) and cooperating companies (Agora Center, University of Jyvaskyla, TeliaSonera, TietoEnator, Metso Automation, Jyvaskyla Science Park) for the grant supporting activities of SmartResource project. We are also grateful to our colleagues Timo Tiihonen, Oleksandr Kononenko and Jouni Pyötsiä for useful discussions and consulting.

References

[Benjamins at al., 2002] R. Benjamins, J. Contreras, O. Corcho and A. Gómez-Pérez, The Six Challenges for the Semantic Web, KR-2002 Workshop on Semantic Web, Toulouse, France, 2002.

[Bizer, 2004] C. Bizer, Trust, Context and Justification, 2004, http://www.wiwiss.fu-berlin.de/suhl/bizer/TrustContextJustification/

[Bouquet, et al., 2004] P. Bouquet, F. Giunchiglia, F. Van Harmelen, L. Serafini, and H. Stuckenschmidt, Contextualizing Ontologies, *Journal of Web Semantics*, Vol. 26, 2004, pp. 1-19.

[Caroll, et al., 2004] J. Caroll, C. Bizer, P. Hayes, and P. Stickler: Named Graphs Provenance and Trust, *Technical Report HPL-2004-57*, Hewlett Packard Research Labs, 2004.

[DAML-Time] Official Homepage of the DAML-Time effort, http://www.cs.rochester.edu/~ferguson/daml/.

[DAML] Official homepage of the DAML program, http://www.daml.org/.

[Hobbs, 2002] J. Hobbs. Towards an ontology of time for the semantic web. In Proc. of Workshop on Annotation Standards for Temporal Information in Natural Language, LREC2002, Las Palmas, Spain, May 2002.

[Hobbs & Pustejovsky, 2003] Jerry R. Hobbs and James Pustejovsky. Annotating and reasoning about time and events. In Patrick Doherty, John McCarthy, and Mary-Anne Williams, editors, *Working Papers of the 2003 AAAI Spring Symposium on Logical Formalization of Commonsense Reasoning*, pages 74-82. AAAI Press, Menlo Park, California, 2003.

[DAML-Time ontology] DAML-Time ontology serialized into OWL/XML: http://www.isi.edu/~pan/damltime/time.owl, http://www.isi.edu/~pan/damltime/time-entry.owl.

[DCMI Metadata Terms] "DCMI Metadata Terms" - DCMI Recommendation, 14 June 2004, http://dublincore.org/documents/dcmi-terms/.

[DublinCore] Official website of the Dublin Core Metadata Initiative, http://dublincore.org/.

[Guha et.al., 2004] R. Guha, R. McCool, and R.Fikes, Contexts for the Semantic Web, In: Proceedings of the Third International Semantic Web Conference (ISWC-2004), Hiroshima, Japan, November 2004, pp. 32-46.

[IOG, 2004] Official Web-Site of Industrial Ontologies Group, http://www.cs.jyu.fi/ai/OntoGroup.

[Jonker et. al., 2001] C. M. Jonker, I. A. Letia, J. Treur, Diagnosis of the Dynamics within an Organization by Trace Checking of Behavioural Requirements. AOSE 2001, pp. 17-32.

[Kaykova et. al., 2004] Kaikova H., Khriyenko O., Kononenko O., Terziyan V., Zharko A., Proactive Self-Maintained Resources in Semantic Web, *Eastern-European Journal of Enterprise Technologies*, Vol. 2, No. 1, 2004, ISSN: 1729-3774, Kharkov, Ukraine, pp. 37-49.

[Khriyenko & Terziyan, 2005] O. Khriyenko, V. Terziyan, Context Description Framework for the Semantic Web, http://www.cs.jyu.fi/ai/papers/Context-2005.pdf.

[Klyne, 2001] G. Klyne, Information Modelling using RDF. Constructs for Modular Description of Complex Systems, March 2001, http://xml.coverpages.org/Klyne-RDFInfoModelling.pdf .

[Klyne & Caroll, 2004] G. Klyne and J. Caroll, Resource Description Framework (RDF) Concepts and Abstract Syntax, W3C Recommendation, http://www.w3.org/TR/2004/REC-rdf-concepts-20040210/, February 2004.

[Manola & Miller, 2004] F. Manola and E. Miller, RDF Primer, W3C Recommendation, http://www.w3.org/TR/2004/REC-rdf-primer-20040210/, February 2004.

[openRDF] Official website of openRDF.org - a community that is the center for all Sesame-related development, http://www.openrdf.org/about.jsp.

[Pan & Hobbs, 2004] F. Pan and J. Hobbs. Time in OWL-S. In Proc. of First International Semantic Web Services Symposium, 2004 AAAI Spring Symposium series, Stanford University, Palo Alto, California, March 22-24, 2004.

[Priebe, 2004] T. Priebe, Context-Based Portlet Integration, Working paper, http://www.inwiss.org/whitepapers/ContextIntegration.pdf, August 2004

[RQL] RQL Tutorial, http://www.openrdf.org/doc/rql-tutorial.html.

[Ryabov & Terziyan, 2003] Ryabov V., Terziyan V., Industrial Diagnostics Using Algebra of Uncertain Temporal Relations, In: *Proceedings of the 21-st IASTED International Multi-Conference on Applied Informatics (AI-2003)*, February 10-13, 2003, Innsbruck, Austria, ACTA Press, ISBN 0-88986-341-5, ISSN 1027-2666, pp.351-356.

[Sayers & Wilkinson, 2003] C. Sayers and K. Wilkinson, A pragmatic Approach to Storing and Distributing RDF in Context using Snippets, Technical Report HPL-2003-231, Hewlett Packard Laboratories, Palo Alto, California, 2003.

[Sheth et al., 2004] A. Sheth, B. Aleman Meza, I. B. Arpinar, C. Ramarkrishnan, C. Bertram, Y. Warke, D. Avant, F. S. Arpinar, K. Anyanwu, K. Kochut, Semantic Association Identification and Knowledge Discovery for National Security Applications, In: Database Technology for Enhancing National Security, *Special Issue of Journal of Database Management*, Vol. 16, No. 1, 2004.

[SmartResource, 2004] Proactive Self-Maintained Resources in Semantic Web, Web Site of SmartResource Tekes Project, http://www.cs.jyu.fi/ai/OntoGroup/SmartResource_details.htm.

[Terziyan & Puuronen, 2000] V. Terziyan, S. Puuronen: Reasoning with Multilevel Contexts in Semantic Metanetworks, In: P. Bonzon, M. Cavalcanti, R. Nossun (Eds.), *Formal Aspects in Context*, Kluwer Academic Publishers, 2000, pp. 107-126.

[Terziyan & Ryabov, 2003] Terziyan V., Ryabov V., Abstract Diagnostics Based on Uncertain Temporal Scenarios, In: M. Mohammadian (ed.), *Proceedings of the International Conference on Computational Intelligence for Modelling Control and Automation (CIMCA-2003),* February 12-14, 2003, ISBN 1740880684, Vienna, Austria, pp. 327-337.

[TimeML] Official website of TimeML project, http://www.cs.brandeis.edu/~jamesp/arda/time/.

[TimeML docs] TimeML documentation, http://www.cs.brandeis.edu/~jamesp/arda/time/timemldocs.html.

APPENDIX

As an example object we use blowing machine with seven main parameters (screw turning speed, open-close stroke, working module pressure, air pressure, oil tank range, oil tank temperature, thermo liquid level), which are taking by seven different sensors. We present two descriptions of the device state:

Device state description (XML-representation):


```
<st:State xmlns:st="http://www.metso.com/Alarm" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
      instance" xsi:schemaLocation=http://www.metso.com/Alarm file:/.../StateMessage.xsd
       Time="2004-09-28T23:50:12.578">
  <Measurement>
    <ParamType>Screw turning speed</ParamType>
    <Units>rpm</Units>
    <Value>70</Value>
    <Sensor sensorID="KS23-S">Rotation speed sensor</Sensor>
  </Measurement>
  <Measurement>
    <ParamType>Open-close stroke</ParamType>
    <Units>mm</Units>
    <Value>70</Value>
    <Sensor sensorID="ll12-D7">Open-close sensor</Sensor>
  </Measurement>
  <Measurement>
    <ParamType>Working module pressure</ParamType>
    <Units>kg/cm2</Units>
    <Value>7.9</Value>
    <Sensor sensorID="Poi1-1">Working module pressure sensor</Sensor>
  </Measurement>
  <Measurement>
    <ParamType>Air pressure</ParamType>
    <Units>kg/cm2</Units>
    <Value>4.6</Value>
    <Sensor sensorID="Pio1-2">Air pressure sensor</Sensor>
  </Measurement>
  <Measurement>
    <ParamType>Oil tank range</ParamType>
    <Units>liter</Units>
    <Value>234</Value>
    <Sensor sensorID="V4-S">Volume measurement sensor</Sensor>
  </Measurement>
  <Measurement>
    <ParamType>Oil tank temperature</ParamType>
    <Units>celcious</Units>
    <Value>62</Value>
    <Sensor sensorID="KX2834-S">Temperature sensor of oil tank</Sensor>
  </Measurement>
  <Measurement>
    <ParamType>Thermo liquid level</ParamType>
    <Units>mm</Units>
    <Value>51</Value>
    <Sensor sensorID="LIQ-23M1">Liquid Sensor</Sensor>
  </Measurement>
</st:State>
```

Device state description (RSCDF-representation):

There is a statement which states that device “#123456XZ24” (assume that instance of this device already exists) has a physical state “#StateContainer181” as a container of the measurements in certain context (in case of container’s context it is a filter statement of the container’s content)…

```
<rdf:Description rdf:about="http://...#StateStatement185">
        <rdf:subject rdf:resource="http://...#123456XZ24" />
        <rdf:object rdf:resource="http://...#StateContainer181" />
        <rscdfs:predicate rdf:resource="http://...ontology#devicePhysicalState" />
        <rdf:type rdf:resource="http://... rscdfs#SR_Statement" />
        <rscdfs:trueInContext rdf:resource="http://...#StateFilterContainer184" />
</rdf:Description>
<rdf:Description rdf:about="http://...#StateContainer181">
        <rscdfs:member rdf:resource="http://...#statement358" />
        <rscdfs:member rdf:resource="http://...#statement210" />
        <rscdfs:member rdf:resource="http://...#statement324" />
        <rscdfs:member rdf:resource="http://...#statement122" />
        <rscdfs:member rdf:resource="http://...#statement80" />
        <rdf:type rdf:resource="http://... rscdfs#SR_Container" />
        <rscdfs:member rdf:resource="http://...#statement64" />
        <rscdfs:member rdf:resource="http://...#statement312" />
</rdf:Description>
<rdf:Description rdf:about="http://...#StateFilterContainer184">
        <rscdfs:member rdf:resource="http://...#stateFilterStatement183" />
        <rdf:type rdf:resource="http://... rscdfs#Context_SR_Container" />
</rdf:Description>
```

Container’s context - filter statement of the container’s content… This statement states that device “#123456XZ24” has some device physical measurement in certain context (time statement “#contextStatement354.time”)…

```
<rdf:Description rdf:about="http://... #stateFilterStatement183">
        <rdf:subject rdf:resource="http://...#123456XZ24" />
        <rdf:type rdf:resource="http://... rscdfs#SR_Statement" />
        <rscdfs:predicate rdf:resource="http://... ontology#devicePhysicalMeasurement" />
        <rscdfs:trueInContext rdf:resource="http://...#FilterStatementContextContainer182" />
</rdf:Description>
<rdf:Description rdf:about="http://...#FilterStatementContextContainer182">
        <rscdfs:member rdf:resource="http://...#contextStatement354.time" />
        <rdf:type rdf:resource="http://... rscdfs#Context_SR_Container" />
</rdf:Description>
```

Following measurement statement states the value (“#numValue355”) for device parameter (“#KF330_param1”), which is a sub property of "#devicePhysicalMeasurement" property, in certain context (“#container357temp”). Numerical value instance “#numValue355” describes parameter value -“70” and measurement unit – “#PromptnessPerMinute”. There are two statements (“#contextStatement356sensorID” – states responsible for measurement sensor; and “#contextStatement354.time” – states time of the environment) which play a role of the context for the measurement statement…

```
<rdf:Description rdf:about="http://...#statement358">
        <rscdfs:predicate rdf:resource="http://... ontology#KF330_param1" />
```

```
        <rdf:object rdf:resource="http://...#numValue355" />
        <rscdfs:trueInContext rdf:resource="http://...#container357temp" />
        <rdf:subject rdf:resource="http://...#123456XZ24" />
        <rdf:type rdf:resource="http://... rscdfs#SR_Statement" />
    </rdf:Description>
    <rdf:Description rdf:about="http://...#numValue355">
        <rscdfs:unit rdf:resource="http://... ontology#PromptnessPerMinute" />
        <rscdfs:value>70</rscdfs:value>
        <rdf:type rdf:resource="http://... rscdfs#NumericalValue" />
    </rdf:Description>
    <rdf:Description rdf:about="http://...#container357temp">
        <rscdfs:member rdf:resource="http://...#contextStatement356sensorID" />
        <rdf:type rdf:resource="http://... rscdfs#Context_SR_Container" />
        <rscdfs:member rdf:resource="http://...#contextStatement354.time" />
    </rdf:Description>
```

Next six sets of the statements describe the rest of the device's measurements (parameters: "#KF330_param2", "#KF330_param3", "#KF330_param4", "#KF330_param5", "#KF330_param6", "#KF330_param7"). All of them are described in the context of correspondent responsible for measurement sensors and the same time statement because all of them are related to the one device state…

```
    <rdf:Description rdf:about="http://...#statement210">
        <rscdfs:trueInContext rdf:resource="http://...#container209temp" />
        <rdf:object rdf:resource="http://...#numValue207" />
        <rdf:type rdf:resource="http://... rscdfs#SR_Statement" />
        <rdf:subject rdf:resource="http://...#123456XZ24" />
        <rscdfs:predicate rdf:resource="http://... ontology#KF330_param2" />
    </rdf:Description>
    <rdf:Description rdf:about="http://...#numValue207">
        <rscdfs:value>70</rscdfs:value>
        <rscdfs:unit rdf:resource="http://... ontology#Millimeter" />
        <rdf:type rdf:resource="http://... rscdfs#NumericalValue" />
    </rdf:Description>
    <rdf:Description rdf:about="http://...#container209temp">
        <rscdfs:member rdf:resource="http://...#contextStatement208sensorID" />
        <rdf:type rdf:resource="http://... rscdfs#Context_SR_Container" />
        <rscdfs:member rdf:resource="http://...#contextStatement354.time" />
    </rdf:Description>

    <rdf:Description rdf:about="http://...#statement324">
        <rscdfs:trueInContext rdf:resource="http://... #container323temp" />
        <rscdfs:predicate rdf:resource="http://... ontology#KF330_param3" />
        <rdf:object rdf:resource="http://...#numValue321" />
        <rdf:subject rdf:resource="http://...#123456XZ24" />
        <rdf:type rdf:resource="http://... rscdfs#SR_Statement" />
    </rdf:Description>
    <rdf:Description rdf:about="http://...#numValue321">
        <rdf:type rdf:resource="http://... rscdfs#NumericalValue" />
        <rscdfs:value>7.9</rscdfs:value>
        <rscdfs:unit rdf:resource="http://... ontology#KilogramPerSquareSantimeter" />
    </rdf:Description>
    <rdf:Description rdf:about="http://...#container323temp">
        <rdf:type rdf:resource="http://... rscdfs#Context_SR_Container" />
        <rscdfs:member rdf:resource="http://...#contextStatement354.time" />
```

```
        <rscdfs:member rdf:resource="http://...#contextStatement322sensorID" />
</rdf:Description>

<rdf:Description rdf:about="http://...#statement122">
        <rscdfs:trueInContext rdf:resource="http://...#container121temp" />
        <rdf:type rdf:resource="http://... rscdfs#SR_Statement" />
        <rdf:subject rdf:resource="http://...#123456XZ24" />
        <rdf:object rdf:resource="http://...#numValue119" />
        <rscdfs:predicate rdf:resource="http://... ontology#KF330_param4” />
</rdf:Description>
<rdf:Description rdf:about="http://...#numValue119">
        <rdf:type rdf:resource="http://... rscdfs#NumericalValue" />
        <rscdfs:unit rdf:resource="http://... ontology#KilogramPerSquareSantimeter" />
        <rscdfs:value>4.6</rscdfs:value>
</rdf:Description>
<rdf:Description rdf:about="http://...#container121temp">
        <rscdfs:member rdf:resource="http://...#contextStatement354.time" />
        <rdf:type rdf:resource="http://... rscdfs#Context_SR_Container" />
        <rscdfs:member rdf:resource="http://...#contextStatement120sensorID" />
</rdf:Description>

<rdf:Description rdf:about="http://...#statement80">
        <rscdfs:trueInContext rdf:resource="http://...#container79temp" />
        <rdf:type rdf:resource="http://... rscdfs#SR_Statement" />
        <rdf:object rdf:resource="http://...#numValue77" />
        <rscdfs:predicate rdf:resource="http://... ontology#KF330_param5" />
        <rdf:subject rdf:resource="http://...#123456XZ24" />
</rdf:Description>
<rdf:Description rdf:about="http://...#numValue77">
        <rscdfs:unit rdf:resource="http://... ontology#Liter" />
        <rscdfs:value>234</rscdfs:value>
        <rdf:type rdf:resource="http://... rscdfs#NumericalValue" />
</rdf:Description>
<rdf:Description rdf:about="http://...#container79temp">
        <rscdfs:member rdf:resource="http://...#contextStatement354.time" />
        <rdf:type rdf:resource="http://... rscdfs#Context_SR_Container" />
        <rscdfs:member rdf:resource="http://...#contextStatement78sensorID" />
</rdf:Description>

<rdf:Description rdf:about="http://...#statement64">
        <rdf:type rdf:resource="http://... rscdfs#SR_Statement" />
        <rscdfs:predicate rdf:resource="http://... ontology#KF330_param6" />
        <rdf:object rdf:resource="http://...#numValue61" />
        <rdf:subject rdf:resource="http://...#123456XZ24" />
        <rscdfs:trueInContext rdf:resource="http://...#container63temp" />
</rdf:Description>
<rdf:Description rdf:about="http://...#numValue61">
        <rdf:type rdf:resource="http://... rscdfs#NumericalValue" />
        <rscdfs:value>62</rscdfs:value>
        <rscdfs:unit rdf:resource="http://... ontology#Celsius" />
</rdf:Description>
<rdf:Description rdf:about="http://...#container63temp">
        <rscdfs:member rdf:resource="http://...#contextStatement62sensorID" />
        <rdf:type rdf:resource="http://... rscdfs#Context_SR_Container" />
        <rscdfs:member rdf:resource="http://...#contextStatement354.time" />
</rdf:Description>
```

```xml
<rdf:Description rdf:about="http://...#statement312">
        <rscdfs:trueInContext rdf:resource="http://...#container311temp" />
        <rdf:subject rdf:resource="http://...#123456XZ24" />
        <rdf:object rdf:resource="http://...#numValue309" />
        <rdf:type rdf:resource="http://... rscdfs#SR_Statement" />
        <rscdfs:predicate rdf:resource="http://... ontology#KF330_param7” />
</rdf:Description>
<rdf:Description rdf:about="http://...#numValue309">
        <rscdfs:value>51</rscdfs:value>
        <rscdfs:unit rdf:resource="http://... ontology#Millimeter" />
        <rdf:type rdf:resource="http://... rscdfs#NumericalValue" />
</rdf:Description>
<rdf:Description rdf:about="http://...#container311temp">
        <rscdfs:member rdf:resource="http://...#contextStatement354.time" />
        <rdf:type rdf:resource="http://... rscdfs#Context_SR_Container" />
        <rscdfs:member rdf:resource="http://...#contextStatement310sensorID" />
</rdf:Description>
```


Following statements define an existence of the sensors (assume that the instances of the sensors already exist), which are parts of the world – “#WorldEnvironment" …


```xml
<rdf:Description rdf:about="http://...#contextStatement356sensorID">
  <rscdfs:predicate rdf:resource="http://... rscdfs#devicePartOfWorld" />
        <rdf:object rdf:resource="http://...#WorldEnvironment" />
        <rdf:subject rdf:resource="http://...#SensorKS23-Sinstance" />
        <rdf:type rdf:resource="http://... rscdfs#SR_Statement" />
</rdf:Description>
<rdf:Description rdf:about="http://...#contextStatement208sensorID">
        <rscdfs:predicate rdf:resource="http://... rscdfs#devicePartOfWorld" />
        <rdf:subject rdf:resource="http://...#SensorII12-D7instance" />
        <rdf:object rdf:resource="http://...#WorldEnvironment" />
        <rdf:type rdf:resource="http://... rscdfs#SR_Statement" />
</rdf:Description>
<rdf:Description rdf:about="http://...#contextStatement322sensorID">
        <rdf:type rdf:resource="http://... rscdfs#SR_Statement" />
        <rdf:subject rdf:resource="http://...#SensorPIO1-1instance" />
        <rscdfs:predicate rdf:resource="http://... rscdfs#devicePartOfWorld" />
        <rdf:object rdf:resource="http://...#WorldEnvironment" />
</rdf:Description>
<rdf:Description rdf:about="http://...#contextStatement120sensorID">
        <rdf:object rdf:resource="http://...#WorldEnvironment" />
        <rdf:type rdf:resource="http://... rscdfs#SR_Statement" />
        <rscdfs:predicate rdf:resource="http://... rscdfs#devicePartOfWorld" />
        <rdf:subject rdf:resource="http://...#SensorPIO1-2instance" />
</rdf:Description>
<rdf:Description rdf:about="http://...#contextStatement78sensorID">
        <rdf:object rdf:resource="http://...#WorldEnvironment" />
        <rscdfs:predicate rdf:resource="http://... rscdfs#devicePartOfWorld" />
        <rdf:type rdf:resource="http://... rscdfs#SR_Statement" />
        <rdf:subject rdf:resource="http://...#SensorV4-Sinstance" />
</rdf:Description>
<rdf:Description rdf:about="http://...#contextStatement62sensorID">
        <rscdfs:predicate rdf:resource="http://... rscdfs#devicePartOfWorld" />
        <rdf:object rdf:resource="http://...#WorldEnvironment" />
```

```
            <rdf:subject rdf:resource="http://...#SensorKX2834-Sinstance" />
            <rdf:type rdf:resource="http://... rscdfs#SR_Statement" />
    </rdf:Description>
    <rdf:Description rdf:about="http://...#contextStatement310sensorID">
            <rdf:type rdf:resource="http://... rscdfs#SR_Statement" />
            <rdf:object rdf:resource="http://...#WorldEnvironment" />
            <rdf:subject rdf:resource="http://...#SensorLIQ-23M1instance" />
            <rscdfs:predicate rdf:resource="http://... rscdfs#devicePartOfWorld" />
    </rdf:Description>
```

Time statement states time for the environment. This statement is used as a context for statements which describe device state and measurements. Temporal instance "#WorldEnvironmentTime353" defines value and unit for time measure.

```
    <rdf:Description rdf:about="http://...#contextStatement354.time">
            <rdf:type rdf:resource="http://... rscdfs#SR_Statement" />
            <rscdfs:predicate rdf:resource="http://... rscdfs#sysTime" />
            <rdf:subject rdf:resource="http://... #WorldEnvironment" />
            <rdf:object rdf:resource="http://...#WorldEnvironmentTime353" />
    </rdf:Description>
    <rdf:Description rdf:about="http://...#WorldEnvironmentTime353">
            <rdf:type rdf:resource="http://... rscdfs#TempTempMark" />
            <rscdfs:value>2004-09-28T23:50:12.578</rscdfs:value>
            <rscdfs:unit rdf:resource="http://... #XMLSchemaDateTime" />
    </rdf:Description>
```

Such definition gives a possibility to describe a hierarchy of nested elements but also refer to any description in flexible way. For example we can easily describe expert's statement about the condition (diagnosis) of the device based on device state sub history. Following statement states diagnosis of the device ("#DeviceDiagnosis_1") in the context of: device state sub history, certain alarm, responsible expert (which states this statement) and time (when this statement was stated)…

```
    <rdf:Description rdf:about="http://...#conditionStatement">
        <rscdfs:trueInContext rdf:resource="http://...#conditionContextStatement" />
        <rscdfs:predicate rdf:resource="http://... ontology#devicePhysicalDiagnosis" />
        <rdf:subject rdf:resource="http://...#123456XZ24" />
        <rdf:object rdf:resource="http://... ontology#DeviceDiagnosis_1" />
        <rdf:type rdf:resource="http://... rscdfs#SR_Statement" />
    </rdf:Description>
    <rdf:Description rdf:about="http://... # conditionContextStatement ">
          <rdf:type rdf:resource="http://... rscdfs#Context_SR_Container" />
          <rscdfs:member rdf:resource="http://... #subHistory" />
         <rscdfs:member rdf:resource="http://... #alarmStatement" />
          <rscdfs:member rdf:resource="http://... #expert1Statement" />
          <rscdfs:member rdf:resource="http://... #timeStatement" />
    </rdf:Description>
```

Helen Kaykova, received her M.Sc. degree in Automated Control Systems in 1982 from Kharkov National University of Radioelectronics (KNURE) in Ukraine. She became Ph.D. in Technical Cybernetics and Information Theory in 1989. She is acting as Docent of the Department of Artificial Intelligence in KNURE and Senior Researcher in Agora Center, University of Jyvaskyla (Finland). Area of research interests includes: Artificial Intelligence: Logic, Temporal Reasoning, AI and Statistics. Helen is a member of the "Industrial Ontologies Group" and takes part in SmartResource Project. (http://www.cs.jyu.fi/ai/helen).

Oleksiy Khriyenko has got his M.Eng. Degree in Intelligent Decision Support Systems in June 2003 from AI Department of KNURE in Ukraine and M.Sc. Degree in Mobile Computing from, Department of Mathematical Information Technology, University of Jyvaskyla in December 2003 from University of Jyvaskyla (Finland). Now he is a postgraduate student on Dept. of Mathematical Information Technology (University of Jyvaskyla, Finland), member of "Industrial Ontologies Group" and researcher in Agora Center (SmartResource project). His research interests include: Artificial Intelligence, Semantic Web, Agent Technology, Web-Services, Context aware Adaptive Environments and Distributed Resource Integration. (http://www.cc.jyu.fi/~olkhriye).

Anton Naumenko has got his M.Sc. Degree in Automated Control Systems in June 2002 from Information Systems Department of KNURE. He is a member of "Industrial Ontologies Group" and researcher in Agora Center ("SmartResource" project). Research interests include: Semantic Web, Agent Technologies, and Role-Based Access Control. (http://people.jyu.fi/~annaumen/).

Vagan Terziyan has got his M.Eng. Degree in Applied Mathematics in 1981, Ph.D. in Technical Cybernetics and Information Theory in 1985 and Doctor of Technical Sciences in 1993 (Dr. Habil Tech. equivalent) in KNURE. He is acting as Professor in Software Engineering and as the Head of Artificial Intelligence Department. Area of research interests includes: Intelligent Web Applications, Distributed AI, Agents, Multiagent Systems and Agent-Oriented Software Engineering, Semantic Web and Web Services, Peer-to-Peer, Knowledge Management, Knowledge Discovery and Machine Learning, Mobile Electronic Commerce. Recently he is working as Associate Professor in MIT Department, University of Jyvaskyla and as Project Leader at the SmartResource TEKES Project in Agora Centre and Head of "Industrial Ontologies Group". (http://www.cs.jyu.fi/ai/vagan).

Andriy Zharko has graduated from Kharkov National University of Radioelectronics in June 2003 with Diploma of Engineer in Intelligent Decision Support Systems. In December 2003 he successfully finished the Master's program at MIT Department, University of Jyvaskyla (Finland), where he obtained Master of Science degree in Mobile Computing. He is member of the "Industrial Ontologies Group" and researcher at the SmartResource Project in Agora Center, University of Jyvaskyla. His research interest is concerned with Peer-to-Peer Semantic Web-based large-scale systems. (http://www.cc.jyu.fi/~anzharko).