Rewiring Strategies for Changing Environments

Wim Laurier*, Geert Vanderhulst°, Geert Poels*, Kris Luyten°

*Department of Management Information and Operational Management, Faculty of Economic and Business Administration, Ghent University, Tweekerkenstraat 2, 9000 Ghent, Belgium
{wim.laurier,geert.poels}@ugent.be

°Hasselt University – transnationale Universiteit Limburg – IBBT Expertise Centre for Digital Media, Wetenschapspark 2, 3590 Diepenbeek, Belgium
{geert.vanderhulst,kris.luyten}@uhasselt.be

Abstract. A typical pervasive application executes in a changing environment: people, computing resources, software services and network connections come and go continuously. A robust pervasive application needs adapt to this changing context as long as there is an appropriate rewiring strategy that guarantees correct behavior. We combine the MERODE modeling methodology with the ReWiRe framework for creating interactive pervasive applications that can cope with changing environments. The core of our approach is a consistent environment model, which is essential to create (re)configurable context-aware pervasive applications. We aggregate different ontologies that provide the required semantics to describe almost any target environment. We present a case study that shows a interactive pervasive application for media access that incorporates parental control on media content and can migrate between devices. The application builds upon models of the run-time environment represented as system states for dedicated rewiring strategies.

Keywords: Dynamic Pervasive Environments, MERODE, ReWiRe, Parental Control

1 Introduction

Mobile devices such as a smart phone or ultra-mobile PC (UMPC) gain popularity and move towards interoperability with pervasive environments. To enable seamless integration of interoperable devices in new environments and to deal with purpose changes (e.g. the role of a mobile phone can evolve from multimedia device to VoIP device), there is a growing need for reconfigurable software applications that can dynamically adapt to their runtime environment [1, 2]. Therefore, this paper presents a combination of ontology-based environment models, which are constructed using the MERODE methodology [3-8] and ReWiRe's environment model [9], which enables software services to react to environment changes (e.g. by. the redistribution of a user interface to another device). Consequently, we rely on the MERODE

methodology to create environment models that go beyond what is needed for creating pervasive environments. The MERODE framework has a strong theoretical underpinning in process algebra and supports dealing with asynchronous and parallel events by which a pervasive environment is characterized. This combination of reconfigurable environment models and pervasive software architecture (i.e. the ReWiRe's technological component is based on OSGi technology[1]) ensures reliable support for building context-aware applications [10] that can be rewired[2] at runtime.
Section 2 introduces the envisioned scenario, where section 3 shows the environment model that is used by the prototype application and section 4 illustrates the prototype application. Conclusions, limitations and directions for future research are discussed in section 5.

2 Scenario: parental control on media content in dynamic pervasive environment

In our envisioned scenario, different types of media are shared on personal devices such as a mobile phone or set-top box with built-in hard disk. When brought together in a connected environment (i.e. a home network), shared media are discovered and listed in a user interface on end-user devices from where they can be accessed and streamed to an output device of choice. For example, an output device could be the current interaction device or a television set capable of playing media (i.e. running a media service). To protect children from content not suited for them (i.e. adult content) parental control is required to successfully deploy a pervasive media application in a real-world environment. Since the environment is assembled at runtime from available devices and media resources, parents have no absolute control over the media resources presented to their children. Therefore, the parental control has to be integrated into the environment model.

This usage scenario demands for a dynamic application that can adapt at runtime to changes in the environment configuration, as changes in the environment configuration can have an impact on the application's execution flow. Consequently, programming such an application in an ad-hoc way would be cumbersome due to the lack of a dynamic knowledge base reflecting the current state of the environment. The impact of environment changes can be indicated considering the example of a television streaming media residing on a mobile phone. If the phone and its owner leave the environment, the media stream is likely to be interrupted and the application will need to cleanup allocated resources (e.g. shutdown a media service and switch off the television screen). Alternatively, when an output device is about to become unavailable, the application might react by automatically selecting another suitable output device to play the current media stream on.

---

[1] http://www.osgi.org/

[2] Rewiring is the dynamic reconfiguration process that enables systems to adapt themselves when their context of use changes.

3 Conceptual Models for Parental Control application

We use the MERODE methodology to create a consistent[3] set of models, which reflect the current environment, and maintain consistency during model changes, which reflect environment changes. A key feature of the MERODE methodology is the use of existence dependency diagrams which relate the existence of a class instance with the existence of one of the instances in its parent class. Such dependency relations mean that instances of the existence dependent class refer to one and the same instance of the parent class during their entire lifespan. Fig. 1 shows the existence dependency diagram for our parental control application. The diagram is represented as a UML class diagram, in which the keywords on top of the class name indicate the origin of the concept that is represented by the class (the RESOURCE class originates in the ReWiRe ontology [9]). Existence dependency relationships are represented as dependency arrows of which the arrowhead points towards the parent class (e.g. NATURALTYPE is a parent class for INDIVIDUAL). The existence dependency semantics are further specified by cardinality constraints added to these arrows. To prevent cognitive overload, methods and attributes were omitted and the 'movie', 'life' and 'person' ontologies are purpose-built minimal ontologies and not fully-fledged ontologies.

The partial ordering of objects, which originates in the existence dependency semantics, determines the object event table, which is a second model (next to the existence dependency diagrams) used by the MERODE methodology and specifies the objects that are affected (i.e. class instances whose state is changed) by the events listed. The matching of objects and events is done at the type level such that the effect that events have on object states can be specified as class methods. MERODE's *event propagation* rule states that the methods that apply to an existence dependent class also apply to its parent class(es), where they can exhibit a different behaviour (i.e. polymorphism). Consequently, an event that creates an object, ends the life of an object or modifies the state of an object may also modify the state of the object's parent object or other objects further up the existence dependency chain. Apart from the existence dependency diagram and the object event table, for each class a lifecycle is specified that describes all possible sequences of object state changes caused by events. These lifecycle models build, together with the existence dependency diagram and the object event table, a conceptual schema for the application.

The conceptual backbone of the reconfigurable conceptual environment model, originates in the work of Parsons and Li [11], who distinguish natural, phase and role types. *Natural types* (e.g. human) are independent (i.e. the existence of their instances does not depend upon other instances) and are rigid (i.e. their individuals cannot migrate to another natural type during their lifetime). *Phase types* (e.g. child, teenager, adult) are independent and anti-rigid (i.e. their individuals can migrate to another phase type during their lifetime). *Role types* are anti-rigid (i.e. their individuals can migrate to another role type during their lifetime) and founded (i.e.

[3] The consistency should prevent children from gaining access to content that is not suitable for them due to weaknesses in the environment model.

depend on a particular pattern of associations). Therefore, individuals are modelled as existence dependent on natural types (cf. rigidity) and phase and role types are modelled as existence dependent of individuals (cf. anti-rigidity).

Using this conceptual backbone, various domain ontology concepts are categorised as natural, phase and role types. Humans are categorised as natural types (i.e. a human individual is a human for all of its life). Also movie types were categorised as natural types. A human's stages of life are considered phases (e.g. being a teenager does not require a particular association and may change to being an adult over time). Furthermore, all concepts of the ReWiRe ontology are considered role types as they all depend on their association with the context of pervasive applications (e.g. USER is the role of a HUMAN INDIVIDUAL in the context of pervasive applications, DEVICE is the role of a piece of hardware in the context of pervasive applications). Consequently, the conceptual backbone enables us to discriminate between the parts of the environment model that can change (i.e. anti-rigid) and those that cannot (i.e. rigid). For example, an adult movie will never be suitable for kids, but a person can grow up. Following the extension of the backbone with domain ontology concepts, the domain ontologies are extended with concepts specific for the intended application, which build an application ontology [12] for parental control. Consequently, the application specific concepts of playing regular media content and adult media content are modelled as extensions of the SERVICE concept in the ReWiRe domain ontology [9].

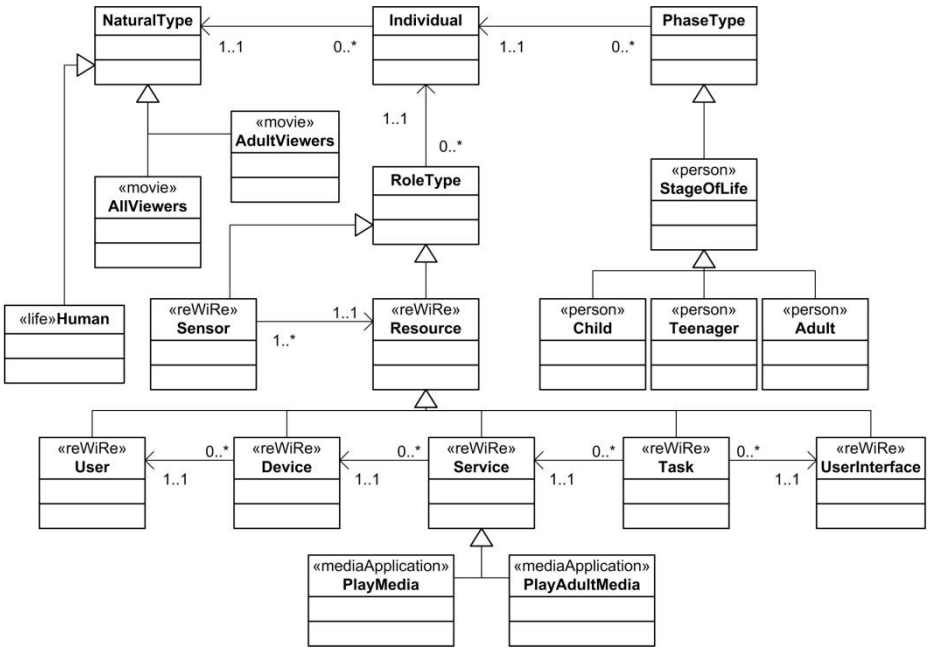

Fig. 1. Conceptual Model of Integrated Ontologies for Parental Control Application

Next to the categorization of domain and application ontology concepts, also the interactions between these concepts in our intended application need to be addressed. These interactions, which we consider as aspects [13] of ontology integration that crosscut multiple ontologies, are modeled as object lifecycle models. Fig. 2 shows one aspect of the interaction between the movie ontology and the media application ontology. It shows that 'all viewer' movies can only be accessed by the 'play media' service. A similar model has been created, representing that 'adult viewer' movies can only be accessed by the 'play adult media' service. The CR_INDIVIDUAL and END_INDIVIDUAL events in the finite state machine indicate that individuals (i.e. movies) can be assigned to each of these natural types.

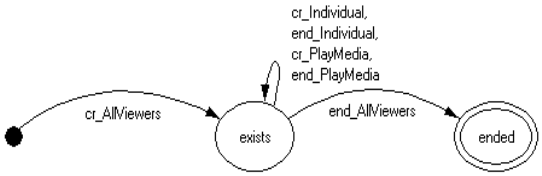

Fig. 2. All Viewers Movie x Play Media Service Aspect

Finally, we discuss the policy model for HUMAN INDIVIDUALS in the context of the parental control application. Fig. 3 shows that only adults (i.e. INDIVIDUALS of natural type HUMAN to which the ADULT phase type has been assigned) can have access to the PLAYADULTMEDIA service (i.e. CR_PLAYADULTMEDIA, end_PLAYADULTMEDIA) where other users only have access to the PLAYMEDIA service (i.e. CR_PLAYMEDIA, end_PLAYMEDIA).

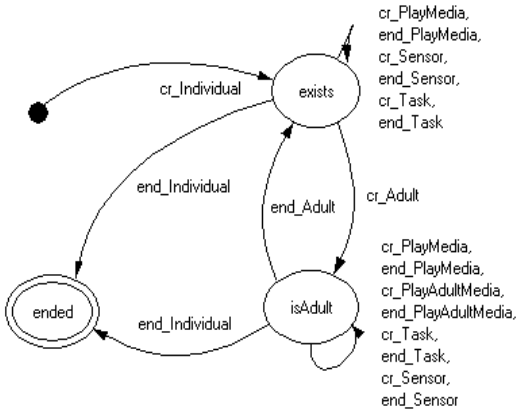

Fig. 3. Adults Only Policy

It should be noted that all finite state machines presented above were simplified for the reason of clarity. However, when these models are represented in the MERODE modeling tool (i.e. Mermaid[4]), the consistency checker automatically identifies e.g. orthogonal events, which can then be added to create the fully-fledged models.

[4] http://merode.econ.kuleuven.ac.be/mermaid.aspx

4 Adopting MERODE models in ReWiRe

The structural models produced by the MERODE design tool are transformed into a media domain ontology which is aggregated with ReWiRe's environment ontology as shown in fig. 4. An instance of this aggregated ontology describes the current environment context (i.e. available resources and relations that apply between them.

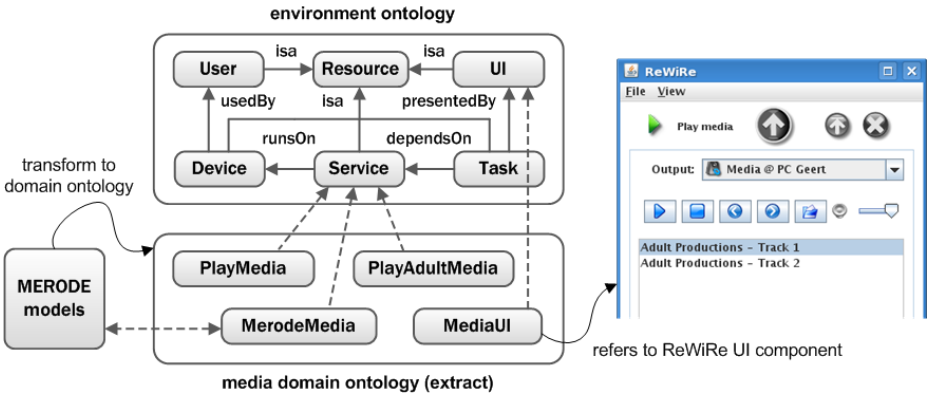

Fig. 4. Aggregated ontologies and media user interface component.

We developed two services and a user interface which, together with the models, give rise to the pervasive media application. The PLAYMEDIA and PLAYADULTMEDIA services implement a media player component that exports a software interface to control a player from external services. The media player component publishes the player's state in the environment model and informs interested parties of state changes through sensor events (using ReWiRe's built-in notification mechanism). The PLAYMEDIA and PLAYADULTMEDIA services also shares media content residing on the device it runs on, by advertising it as MEDIACONTENT in the environment model. An additional service, the MERODEMEDIA service, fulfils a coordination role and implements a dedicated 'rewiring strategy' for the pervasive media scenario. Such a rewiring strategy aims to keep an application consistent at all times, in particular when the configuration of the environment changes. The coordination service is built on top of our previously designed MERODE models. As such, the objects generated from the structural and behavioural knowledge base serve as a special-purpose model which is synchronized with the ReWiRe environment model using asynchronous events. For instance, the coordination service subscribes to R+ events (a new resource enters the environment), R- events (a resource leaves the environment) and player state events fired by a PLAYMEDIA or PLAYADULTMEDIA service and propagates this information to the MERODE models. Modifications to these models (e.g. transformations from one state into another one), either triggered by system or user events, are then translated into method calls that orchestrate a play media service.

Furthermore, we developed and deployed a migratable media user interface component as shown in figure 4. This interface presents available media content on

end-user devices and features the option to select an output device to stream selected media to. In its back-end, this user interface leverages the MERODEMEDIA coordination service and its embedded behaviour models. For example, if a user selects adult content, MERODE models are traversed and if an adult check does not pass, the play button remains disabled in the user interface for the selected media.

5 Conclusions, limitations and Future Research

This paper presented a framework, which uses ontology models that are connected to a conceptual backbone to provide pervasive applications with a shared and dynamic environment model, of which the (run-time) consistency is guaranteed by using the MERODE methodology. The presented parental control application is one of many potential scenarios that requires both real-world (e.g. the age of a person, the rating of a movie) and technological awareness (e.g. listing potential output devices). The integration of real-world and technological awareness in a changing (real-world and technological) environment is enabled by combining MERODE and ReWiRe.

Models that can capture the current state of a pervasive environment and its applications are vital components to build context-aware applications [14]. Ontologies have already been used to enable the development of pervasive applications [10, 15, 16]. For example, Preuveneers et al. proposed an ontology to capture the context of use of ambient intelligent environments in [17]. Chen et al. [15] designed a rich ontology for ubiquitous and pervasive applications (SOUPA) that is exploited in a broker-centric agent architecture to support knowledge sharing and context reasoning. In [10] context discovery and knowledge sharing are supported using an ontology-based context model and an OSGi-based middleware infrastructure. However, their ontology is mainly geared towards the creation and deployment of context-aware systems, and does not focus on runtime and thus (re)configuration support. As we do not only use ontologies to capture and query the execution context, but also to create dynamic pervasive applications which can be (re)configured at runtime. As the MERODE algebra has been used in model-driven design, it should be noted that the framework presented in this paper differs from modelling languages for pervasive systems (e.g. PervML[5]) by discriminating a dynamic environment model, of which the consistency is supported by the MERODE algebra, and a stable application design in ReWiRe, which interprets the environment model, where model-driven design approaches focus on the application design.

Since the use of the MERODE algebra has been limited to model-driven design, a tool that integrates the MERODE methodology and the ReWiRe framework is not available yet. Consequently, part of the integration between the MERODE and the ReWiRe environment had to be performed manually. In the future, the MERODE algebra will be integrated in the ReWiRe framework to automate environment model reconfiguration fully.

[5]http://oomethod.dsic.upv.es/labs/index.php?option=com_content&task=view&id=40&Itemid=77

References

1. Geihs, K., Barone, P., Eliassen, F., Floch, J., Fricke, R., Gjorven, E., Hallsteinsen, S., Horn, G., Khan, M.U., Mamelli, A., Papadopoulos, G.A., Paspallis, N., Reichle, R., Stav, E.: A comprehensive solution for application-level adaptation. Softw. Pract. Exper. **39** (2009) 385-422
2. Lobato, C., Garcia, A., Romanovsky, A., Lucena, C.: An aspect-oriented software architecture for code mobility. Softw. Pract. Exper. **38** (2008) 1365-1392
3. Snoeck, M., Lemahieu, W., Goethals, F., Dedene, G., Vandenbulcke, J.: Events as atomic contracts for component integration. Data & Knowledge Engineering **51** (2004) 81-107
4. Snoeck, M., Dedene, G.: Existence dependency: The key to semantic integrity between structural and behavioral aspects of object types. IEEE Transactions on Software Engineering **24** (1998) 233-251
5. Dedene, G., Snoeck, M.: Formal deadlock elimination in an object oriented conceptual schema. Data & Knowledge Engineering **15** (1995) 1-30
6. Snoeck, M., Dedene, G.: Generalization/specialization and role in object oriented conceptual modeling. Data & Knowledge Engineering **19** (1996) 171-195
7. Snoeck, M., Poels, G.: Improving the Reuse Possibilities of the Behavioral Aspects of Object-Oriented Domain Models. 19th International Conference on Conceptual Modeling — ER 2000, Vol. 1920. Springer (2000) 423-439
8. Snoeck, M.: Object-oriented enterprise modelling with MERODE. Leuven University Press, Leuven (1999)
9. Vanderhulst, G., Luyten, K., Coninx, K.: ReWiRe: Creating interactive pervasive systems that cope with changing environments by rewiring. 4th International Conference on Inteligent Environments. IEEE, Seattle, WA (2008) 1-8
10. Gu, T., Pung, H.K., Zhang, D.Q.: Toward an OSGi-Based Infrastructure for Context-Aware Applications. IEEE Pervasive Computing **3** (2004) 66-74
11. Parsons, J., Li, X.: An ontological Metamodel of Classifiers and Its Application to Conceptual Modelling and Database Design. 26th International Conference on Conceptual Modeling (ER 2007), Vol. 4801 (2007) 214-228
12. Guarino, N.: Formal Ontology and Information Systems. Proceedings of FOIS'98. IOS Press, Trento, Italy (1998) 3-15
13. Kiczales, G., Lamping, J., Mendhekar, A., Maeda, C., Lopes, C., Loingtier, J.M., Irwin, J.: Aspect-oriented programming. In: Aksit, M., Matsuoka, S. (eds.), Jyvaskyla, Finland (1997) 220-242
14. Dey, A.K.: Understanding and Using Context. Personal and Ubiquitous Computing **5** (2001) 4-7
15. Chen, H., Perich, F., Finin, T., Joshi, A.: SOUPA: standard ontology for ubiquitous and pervasive applications. Mobile and Ubiquitous Systems: Networking and Services, 2004. MOBIQUITOUS 2004. The First Annual International Conference on (2004) 258-267
16. Peters, S., Shrobe, H.E.: Using semantic networks for knowledge representation in an intelligent environment. First IEEE International Conference on Pervasive Computing and Communications IEEE (2003) 323-329
17. Preuveneers, D., Van den Bergh, J., Wagelaar, D., Georges, A., Rigole, P., Clerckx, T., Berbers, Y., Coninx, K., Jonckers, V., De Bosschere, K.: Towards an Extensible Context Ontology for Ambient Intelligence. Ambient Intelligence (2004) 148-159