Assisting the User in Selecting Devices for Media Content

Florian Fuchs, Iris Hochstatter, and Sebastian Henrici
Mobile and Distributed Systems Group, Ludwig-Maximilians-Universität München
{florian.fuchs|iris.hochstatter}@ifi.lmu.de, sebastian.henrici@cip.ifi.lmu.de

Abstract: Ubiquitous computing environments have to assist users in choosing the right device for any given media content they want to consume. Currently, there are few existing approaches that match the content features with available devices and still let the user have control over the process. In this paper, we propose a new approach to this problem based on ontological modeling and reasoning. Context information concerning available devices and content is taken into account to automatically derive appropriate devices for a particular media content. We combine this semantic matching with policy-based control in order to achieve a trade-off between automation and user control.

1 Introduction

Ubiquitous Computing environments and user mobility will change the way people consume media content: Instead of being restricted to one device (usually the one the content is stored on), there will be a multitude of other devices available in the user's current environment. Some of these devices will probably be better suited to present a particular media content than the device the content is stored on. Consider for example a video clip that is stored on a PDA. Instead of watching the clip on the small screen of the PDA, the user might prefer to have it presented on the larger screen of the nearby LCD.

However, it is burdensome for the user to be prompted to select a device from the list of all currently available devices every time a media content is to be played. In rich intelligent environments, the length of the device list may also be overwhelming. This is why the user requires assistance in selecting the appropriate device for playing a given media content. At the same time, the user should still remain in control of the content provisioning process in order to be able to adapt it to personal preferences. As a consequence, an appropriate trade-off between automation and user control has to be achieved.

Consider the following scenario: *Bob is on a business trip and enters his hotel room, which is equipped with a large TV set, a computer monitor and speakers. He decides to read some of the emails he downloaded to his PDA earlier that day in the office. The first email is from his friend Bill consisting of a video of their skiing trip and a short text notice. It is automatically derived that of all the available devices, only the PDA, the TV set and the computer monitor are capable of displaying the text message. As Bob prefers the small PDA display when reading messages containing a short text, the text message is displayed on his PDA. Concerning the video clip it is automatically derived that only the TV set and the computer monitor are capable of displaying it. As Bob set in his profile that he wants* *videos to be played on the largest screen available, the video message is shown on the TV.*

This simple scenario implies that automating content/device matching requires an expressive model of both the content and devices. It has to cover technical features of content as well as additional features such as legal aspects. The model has to be flexibly extensible in order to incorporate newly emerging devices as well as new content types and features. It has to support independent extensions by different parties, as many different stakeholders will be involved. At the same time, the matching process should be generic in order to automatically deal with new devices and media content without requiring adaptation.

There are few existing approaches which actually take into account content features when discovering devices and still let the user have control over the process. Traditional service discovery systems such as UPnP (http://www.upnp.org) or HAVi (http://www.havi.org) rely on simple and fixed models for the content. This limits flexibility and precision of the matching and also hampers extension of the matching process for new devices and media contents. Moreover, user preferences cannot be taken into account at all. Semantic service discovery (e.g. [KLP$^+$04]) gained a lot of interest recently, but is generally geared towards web services and does not consider media content and devices. [MRCM03] presents work on semantic discovery in Ubiquitous Computing environments. Still, these approaches do consider the characteristics of media content. Models of devices and media content are already available, but they are not used for content/device matching: MIME [FB96] only identifies content formats as it was developed to transfer data. The MPEG-7 standard [NL99] and the aceMedia project [KAHS04] aim at modeling both subject matter and technical properties of content, but do not consider devices. The FIPA Device Ontology (http://www.fipa.org) describes some devices, but provides only coarse-grained models.

In this paper, we present an ontology-based approach for assisting the user in selecting an appropriate device for a given media content. Here, we focus on the selection process and abstract from particular middleware (e.g. communication, protocols) and presentation aspects (e.g. streaming, synchronization) . Our approach provides an expressive and extensible model of content and devices as a basis for formulating matching criteria using generic terms. New types of content and devices can easily be incorporated at a later stage without requiring adaptation of the matching process. In order to balance autonomation and user control, this is combined with policies expressing user preferences.

The paper is structured as follows: Section 2 proposes a three-step approach for balancing automation and user control in the content provisioning process. Section 3 presents our approach to semantic matching based on ontological modeling and reasoning. In Section 4 we describe our prototype implementation, which shows the feasibility of our approach. Section 5 concludes the paper.

2 Balancing Automation and User Control for Device Selection

We propose a three-step process for assisting the user in selecting the most appropriate device for a given content. It combines both automated steps using ontology-based matching and user-controlled steps based on user policies:

Step 1: Initiation Based on User Preferences. This step decides whether devices in the surrounding should be considered for playing a particular type of media content. Users state their preferences as policies with respect to content features. One example is that a video should always be played on the largest screen available. The process either terminates or continues with step two.

Step 2: Prefiltering Based on Automated Semantic Matching. This step uses semantic matching (see Section 3) in order to identify for the given media content the appropriate devices out of all available devices. For example, the description of a video content would be matched with the descriptions of surrounding devices and a TV set could be derived to be applicable. This step reduces the number of potential device choices for the user and results in a list of appropriate devices.

Step 3: Final Device Selection Based on User Preferences. This step applies user preferences in order to personalize device selection. Here, users state their preferences as policies with respect to device features. For example, even though a computer screen is a potential match for video content, the user can state that the device with the largest screen should always be selected for video content. This step results in the final list of appropriate devices for the given content, which is then presented to the user.

3 Semantic Matching based on Ontological Modeling and Reasoning

This section details step two of the process, which uses ontological modeling and reasoning for automatic matching of contents and devices. Ontologies are a tool for knowledge representation that specify a shared understanding of concepts, relations and their instances in a particular domain. They can be specified using Description Logics (DL) [BCM$^+$03], which allow to capture and infer certain semantics such as subsumption and classification in a machine-interpretable way. Supporting consistency checks, they can easily be extended. These properties make DL-based ontologies a promising tool for realizing semantic matching of contents and devices.

We first introduce the proposed upper ontology that integrates content domain and device domain. Then we elaborate on how to use reasoning on this representation in order to perform semantic matching.

Modeling Devices and Contents. The basic idea of our approach is to provide an integrated ontology for both content and device modeling, which can easily be detailed with specific content and device types at a later stage. Figure 1 shows the proposed upper ontology containing basic concepts for both the content and the device domain and linking them via appropriate relations.

The fundamental concept within the **content domain** is the *Content* concept. An instance of the *Content* concept represents an aggregate of content as a whole and can be associated with general properties such as the duration of the content, the creator of the content or the overall cost connected with the content. Actual pieces of content such as video or

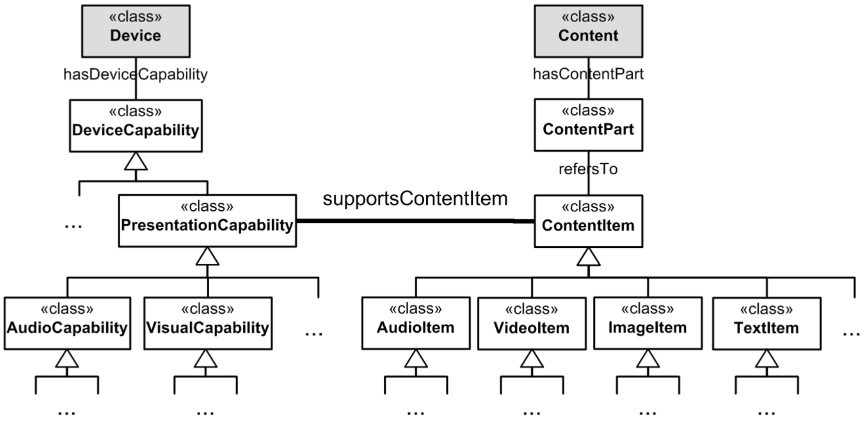

Figure 1: Integrated content and device ontology (illustrated in UML).

audio clips are represented as instances of the *ContentItem* concept. The link between the *Content* concept and the *ContentItem* concept is provided by the *ContentPart* concept: Instances of *Content* refer to instances of *ContentPart*, this way specifying the temporal structure of the *Content* instance. Instances of *ContentPart*, in turn, refer to instances of *ContentItem*. The *ContentItem* concept can further be specialized to *AudioItem*, *VideoItem*, *ImageItem*, *TextItem*, and so on. In the scenario, the email as an aggregate of content is represented by a *Content* instance. It is related to two *ContentPart* instances, which reference the relevant sections of the text content and the video content. Finally, the actual text content and video content are represented by instances of the two *ContentItem* subclasses *TextItem* and *VideoItem*, respectively.

The fundamental concept within the **device domain** is the *Device* concept. It serves as the abstract concept for a comprehensive, extensible device taxonomy. A particular *Device* is described using instances of the *DeviceCapability* concept. *DeviceCapability* is among others further specialized to *PresentationCapability*, which represents the capability of presenting a particular type of content. For example, a TV set could be represented as an instance of the *Device* concept associated with instances of the specialized *Presentation-Capability* concepts *VisualCapability* and *AudioCapability*.

The link between **the content domain and the device domain** is provided by the *supportsContentItem* relations, which associates the *PresentationCapability* concept with the *ContentItem* concept. In the TV set example, the TV's *VisualCapability* could be related via *supportsContentItem* with the *ContentItem* specializations *VideoItem*, *ImageItem*, and *TextItem*. In addition, the TV's *AudioCapability* could be associated with *AudioItem*.

Semantic Matching based on Reasoning. For realizing semantic matching, ontological reasoning is applied to the previously described content and device ontology. We assume that information about available *devices* is acquired externally (similar to UPnP each device could provide its own device description) and represented as instances of this ontology in a knowledge base. We also assume that information about the relevant *content* (e.g. generated from existing meta-information such as MIME type) is represented as instances of this ontology in the same knowledge base. This information being available, the problem of selecting appropriate devices is therefore abstracted to inferring the list of device instances with capabilities matching the content instance.

Matchmaking queries are used for specifying the matching criteria in a declarative way. They are formulated with respect to the content and device ontology. All device instances satisfying a matchmaking query therefore represent appropriate devices. A matchmaking query is formalized as $(c_1\ ,\ c_2\ ,\ ...\ ,\ c_n)$ WHERE $(x_1\ ,\ x_2\ ,\ ...\ ,\ x_n)$.

$c_1, c_2, ..., c_m$ are *atoms* that contain variables and $x_1, x_2, ..., x_m$ are *variable bindings* that refer to variables on the left hand side. These variable bindings allow to parameterize the matchmaking query at runtime. Several atoms or variables are treated as a conjunction. Atoms are of the form *C(i)*, *direct(C,i)* and *P(i1, i2)*: *C(i)* is a class atom, where *C* is either a class from the ontology or a class variable; *i* is either an instance or an instance variable. *C(i)* holds if *i* is an instance of class *C* or one of its specializations. *direct(C,i)* holds if *i* is a *direct* instance of class *C*. *P(i1, i2)* is a property atom, where *P* is a property from the ontology or a property variable. *i1* and *i2* are instances or instance variables. *P(i1, i2)* holds if *i1* and *i2* are instances of the domain and range definition of property *P*.

The following sample matchmaking query identifies devices with appropriate presentation capabilities for the given video clip. It is evaluated with respect to the current state of the knowledge base. Discovered bindings for the `dev` variable represent matching devices:

```
(ContentItem(item), direct(itemType,item),
 Device(dev), PresentationCapability(deviceCap),
 hasDeviceCapability(deviceCap,cap),
 supportsContentItem(cap,item2), direct(itemType,item2))
 WHERE (item = videoClip53)
```

Note that the query can be formulated at design time and uses only generic terms from the upper ontology. Still, when executing this query at run time, the reasoning procedure additionally takes into account instances of all specializations of the used ontology terms. This way it is not necessary to explicitly list all device types that might be potential matches. Instead, the matching criteria are formulated with respect to content and device features in a declarative way and can easily be adapted. Due to the ontological model, even previously unknown device types and content features are automatically taken into account without requiring modification of the query. These properties are especially relevant in intelligent environments which comprise of a multitude of devices and are highly dynamic.

4 Prototype Implementation

We realized our approach by implementing the three steps of the proposed content provisioning process. Special emphasis was given to the semantic matching procedure. As a first step for evaluating the feasibility of our approach, we also implementated an agent-based simulation of the presented scenario.

Device Selection Process. Steps one and three of the process evaluate user preferences. Preferences are stated as rules with respect to the concepts specified in the ontology. They were implemented using the rule engine JESS [FH03]. The following rule states that short notes should be displayed on the user's PDA (namespaces are abbreviated; *Note* is a content property defined in the ontology as consisting of at most 20 characters):

```
(defrule rule-for-short-note
(triple (predicate "rdf:type")
        (subject ?q)
        (object "ont:TextItem"))
(triple (predicate "ont:hasProperty")
        (subject ?q)
        (object ?o))
(triple (predicate "rdf:type")
        (subject ?o)
        (object "ont:Note")) => (present-decision "ont:PDA"))
```

Step two requires ontology processing. For specifying the content and device ontology we used OWL DL, the Description Logic (DL) variant of the Web Ontology Language OWL [MvH04]. Matchmaking queries are executed using the Semantic Web toolkit Jena2 (http://jena.sourceforge.net) connected to the open-source DL reasoner Pellet [SPG$^+$05]. Matchmaking queries are translated into a combination of both instance level and concept level queries. As there is still no standardized query language supporting both kinds of queries, we used a combination of SPARQL queries [PS05] for the instance level and Pellet's Java interface for the concept level.

Scenario Simulation. As a framework for simulating the relevant components of the device selection process we used the Java Agent Development Framework JADE (http://jade.tilab.com), and implemented the following agents:

- *DeviceAgent* represents a particular device. It provides a device description with respect to the content and device ontology and registers it with a *DeviceBrokerAgent*. Each device from the scenario, such as the TV set, is represented by a *DeviceAgent*.
- *DeviceBrokerAgent* maintains a directory of device agents and their descriptions. For the scenario, this agent represents the intelligent environment infrastructure.
- *ContentProvisioningAgent* controls the overall process of device selection. If the user policies on content types evaluate to initiating the semantic matching, it passes the content description to the *SemanticMatchingAgent*, which in turn returns the list of appropriate devices. It then evaluates the user policies on device types and presents the final device selection to the user. For the scenario, this agent represents the device selection engine on the PDA.
- *SemanticMatchingAgent* performs the semantic matching of content and devices. It periodically updates its knowledge base of available devices by querying a *DeviceBrokerAgent*. When receiving a content description from the *ContentProvisioningAgent*, it adds this description to the knowledge base and executes the matchmaking query using the reasoning service of Pellet. All matching device instances are returned. For the scenario, this agent would be located on the PDA.

5 Conclusion and Future Work

In this paper, we described a novel approach to assisting the user in selecting devices for media content. We focused on the selection process and proposed to partially automate it using semantic matching based on ontological modeling and reasoning. Together with an integrated content and device ontology we introduced the notion of matchmaking queries for inferring appropriate devices given a particular content. In order to balance automation and user control we combined semantic matching with policy-based control and proposed a three-step process. A prototype showed the feasibility of our approach and simulated our sample scenario.

Future work includes incorporating additional context information into the selection process such as the current location and social setting of the user. Furthermore, we plan to deploy the proposed selection process on an existing service discovery middleware such as UPnP in order to evaluate it in a field study.

References

[BCM$^+$03] Franz Baader, Diego Calvanese, Deborah McGuinness, Daniele Nardi, and Peter F. Patel-Schneider, editors. *The Description Logic Handbook: Theory, Implementation, and Applications*. Cambridge University Press, 2003.

[FB96] N. Freed and N. Borenstein. Multipurpose Internet Mail Extensions (MIME). RFC2045: Internet (Network Working Group), November 1996.

[FH03] Ernest Friedman-Hill. *Jess in Action: Java Rule-based Systems*. Manning Publications Co., 2003.

[KAHS04] Yiannis Kompatsiaris, Yannis Avrithis, Paola Hobson, and Michael Strintzis. Integrating Knowledge, Semantics and Content for User-Centered Intelligent Media Services: The aceMedia Project. In *Proc. of Workshop on Image Analysis for Multimedia Interactive Services*, 2004.

[KLP$^+$04] Michael Kifer, Ruben Lara, Axel Polleres, Chang Zhao, Uwe Keller, Holger Lausen, and Dieter Fensel. A Logical Framwork for Web Service Discovery. In *Proc. of Semantic Web Services Workshop at 3rd Int'l Semantic Web Conference*, November 2004.

[MRCM03] Robert McGrath, Anand Renganathan, Roy Campbell, and Dennis Mickunas. Incorporating "Semantic Discovery" into Ubiquitous Computing Infrastructure. In *Proc. of System Support for Ubiquitous Computing Workshop at UbiComp*, October 2003.

[MvH04] Deborah L. McGuinness and Frank van Harmelen. OWL Web Ontology Language Overview. W3C Recommendation, 2004.

[NL99] Frank Nack and Adam T. Lindsay. Everything you wanted to know about MPEG-7: Part 1 and 2. *IEEE MultMedia*, 3:65–77, 1999.

[PS05] Eric Prud'hommeaux and Andy Seaborne. SPARQL Query Language for RDF. W3C Working Draft, November 2005.

[SPG$^+$05] Evren Sirin, Bijan Parsia, Bernardo Cuenca Grau, Aditya Kalyanpur, and Yarden Katz. Pellet: A Practical OWL-DL Reasoner, 2005. Submitted to Journal of Web Semantics.