Epimenides: An Information System offering Automated Reasoning for the Needs of Digital Preservation

Yannis Kargakis and Yannis Tzitzikas
Institute of Computer Science, FORTH-ICS
Computer Science Department, University of Crete, Greece
{kargakis|tzitzik}@ics.forth.gr

ABSTRACT

`Epimenides` is a system that can be used in the context of digital archives and digital libraries for helping archivists in checking whether the archived digital artifacts remain *intelligible* and *functional*, and in identifying the consequences of probable losses. A distinctive feature of `Epimenides` is that it can model also *converters* and *emulators*, and the adopted modelling approach enables the *automatic reasoning* needed for reducing the human effort required for checking whether a task can be performed over a digital object (or digital collection in general).

1. INTRODUCTION AND MOTIVATION

Digital material has to be preserved not only against loss or corruption, but also against changes in its ecosystem for ensuring its usability. The latter is essentially a dependency management problem. Since *conversion* (else migration) and emulation are fundamental preservation strategies, a dependency management approach for digital preservation should be able to model converters and emulators, and also to exploit their capabilities. This is of paramount importance since a sequence of conversions and emulations can be enough to vanish the gap that prevents performing a task on a digital object.

Since there are numerous frameworks, tools and approaches for the emulation of computer systems (hardware and software components) and the conversion of file formats, it is practically impossible for a human to exploit all the possible ways that can arise from their capabilities for performing a task. For this reason, and in order to assist the *preservation planning*, we propose using advanced knowledge management techniques.

`Epimenides`[1] is the first system that realizes the approach presented in [1] and offers automated reasoning for (a) *Task-Performability Checking*, (b) *Consequences of a Hypothetical Loss* and (c) *Identification of Missing Modules*. A Use Case Diagram providing an overview of the supported use cases is given in Figure 1.

[1]www.ics.forth.gr/isl/epimenides/

DL'14, September, 2014, London, UK.

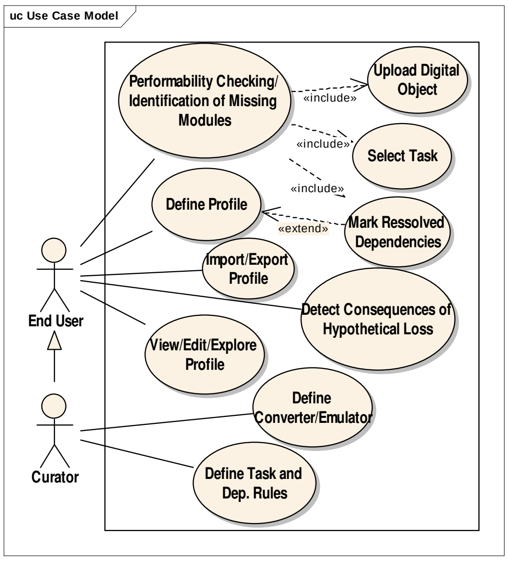
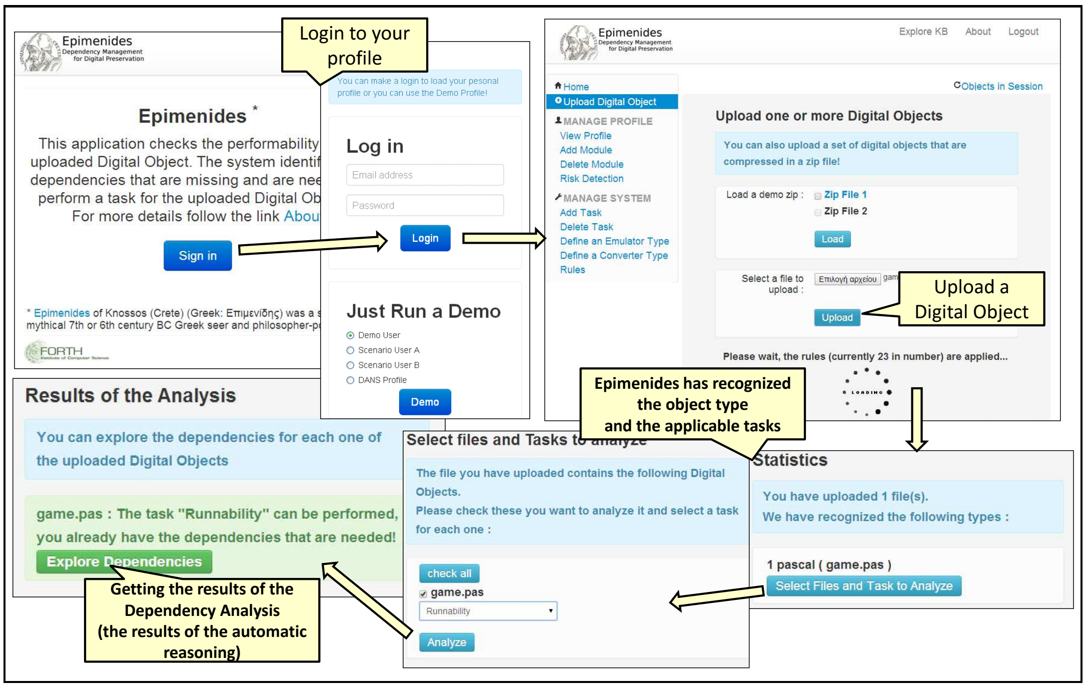

uc Use Case Model
Performability Checking/ Identification of Missing Modules
Upload Digital Object
«include»
«include»
Select Task
«include»
Define Profile
«extend»
Mark Ressolved Dependencies
Import/Export Profile
Detect Consequences of Hypothetical Loss
End User
View/Edit/Explore Profile
Define Converter/Emulator
Curator
Define Task and Dep. Rules

Figure 1: Use Case Diagram of `Epimenides`

2. DESCRIPTION OF THE SYSTEM

We can convey the main message through an example. Consider a user, say Yannis, who would like to compile and run on his mobile phone, software source code written before many years, e.g. software code written in Pascal programming language, stored in a file named `game.pas`. For example consider the situation illustrated in Figure 2a. The rising questions are: *what can Yannis do (to achieve his objective), what should we (as community) do, do we have to develop a Pascal compiler for Android OS, do we have to standardize programming languages*? The direction and answer of the above questions (according to the approach that `Epimenides` follows), is that it is worth investigating whether it is already possible to compile and run that code on android by "combining" existing software, i.e. by applying a series of transformations and emulations. To continue this example, suppose that we have in our disposal only the modules (as a *module* we consider a software or hardware component) that are shown in Figure 2b. Someone could then think that it seems that we could run `game.pas` on Yannis' mobile phone in three steps: by first converting the Pascal code to C++ code, then compiling the C++ code


Login to your profile

Epimenides

This application checks the performability uploaded Digital Object. The system identif dependencies that are missing and are nee perform a task for the uploaded Digital Ob For more details follow the link Abou

Sign in

* Epimenides of Knossos (Crete) (Greek: Επιμενίδης) was a s mythical 7th or 6th century BC Greek seer and philosopher-p

You can make a login to load your pesonal profile or you can use the Demo Profile!

Log in

Email address

Password

Login

Just Run a Demo

Demo User

Scenario User A

Scenario User B

DANS Profile

Demo

Explore KB About Logout

Home

Upload Digital Object

MANAGE PROFILE

View Profile

Add Module

Delete Module

Risk Detection

MANAGE SYSTEM

Add Task

Delete Task

Define an Emulator Type

Define a Converter Type

Rules

Objects in Session

Upload one or more Digital Objects

You can also upload a set of digital objects that are compressed in a zip file!

Load a demo zip : Zip File 1 Zip File 2

Load

Select a file to upload : Επιλογή αρχείου

Upload

Upload a Digital Object

Please wait, the rules (currently 23 in number) are applied...

Epimenides has recognized the object type and the applicable tasks

Statistics

You have uploaded 1 file(s).

We have recognized the following types :

1 pascal (game.pas)

Select Files and Task to Analyze

Select files and Tasks to analyze

The file you have uploaded contains the following Digital Objects.

Please check these you want to analyze it and select a task for each one :

check all

game.pas

Runnability

Analyze

Results of the Analysis

You can explore the dependencies for each one of the uploaded Digital Objects

game.pas : The task "Runnability" can be performed, you already have the dependencies that are needed!

Explore Dependencies

Getting the results of the Dependency Analysis (the results of the automatic reasoning)

Figure 3: Checking the performability over a digital object

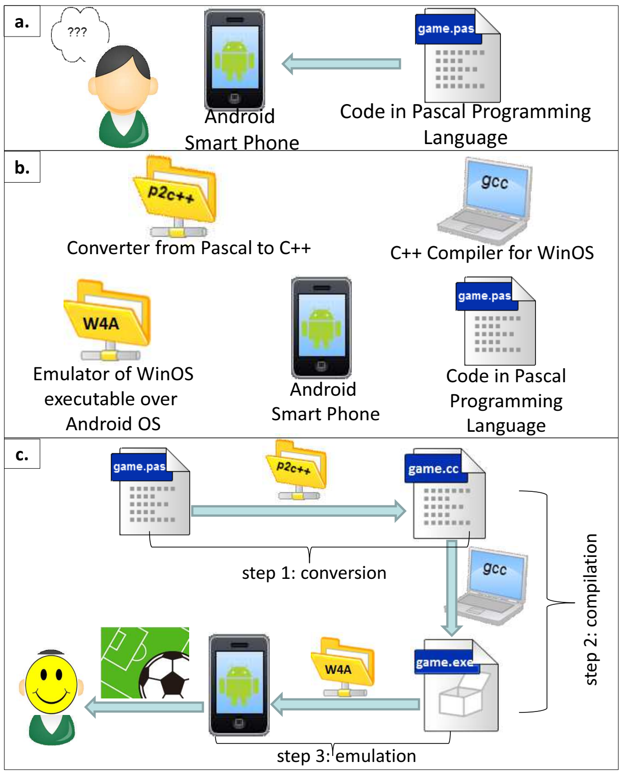


Figure 2: Running example. (a) The problem, (b) The available modules, (c) A series of conversions/emulations to achieve our objective

to produce executable code, and finally by running over the emulator the executable yielded by the compilation. Indeed, the series of transformations/emulations shown in Figure 2c could achieve our objective. However, one might argue that this is very complex for humans. Indeed this is true, and this is the reason why we believe that such reasoning should be done by computers, not humans. `Epimenides` enables this kind of *automated reasoning*. Figure 3 summarizes the interaction between a user and `Epimenides` for checking the `Runnability` over `game.pas`.

`Epimenides` is based on W3C standards, and its Knowledge Base (KB), expressed using RDF/S, currently contains information about 657 Module Types, including 647 MIME Type Modules. The KB contains information about three tasks (readability, runnability, rendering) and for their dependencies 53 rules have been specified. In total, the KB contains 2,304 RDF triples. As regards extensibility, any user can enrich the KB by adding his/her own Module Types, Tasks and Rules, and since the KB is based on Semantic Web technologies it can be straightforwardly enriched with information coming from other external sources (i.e. other SPARQL endpoints). `Epimenides` offers a web interface, the server side uses the Apache Tomcat[2] 7.0.3 web server and the triple store OpenLink Virtuoso[3] 06.01.3127 version, while the Virtuoso Jena RDF Data Provider[4] is used for the communication with the triplestore.

Acknowledgements

`Epimenides` was partially supported by the APARSEN Network of Excellence (FP7-ICT-2009-6 agreement 269977).

3. REFERENCES

[1] Y. Tzitzikas, Y. Marketakis, and Y. Kargakis, "Conversion and Emulation-aware Dependency Reasoning for Curation Services," in *Proceedings of the 9th Annual International Conference on Digital Preservation (iPres2012)*, 2012.

[2]http://tomcat.apache.org/
[3]http://virtuoso.openlinksw.com/
[4]http://www.openlinksw.com/dataspace/doc/dav/wiki/Main/VirtJenaProvider