Project Number	IST-2006-033789
Project Title	Planets
Title of Deliverable	**Planets Planning Tool (version 3)**
Deliverable Number	**PP4/D5**
Contributing Sub-project and Work-package	PP4
Deliverable Dissemination Level	External Public
Deliverable Nature	Report
Contractual Delivery Date	31st May 2010
Actual Delivery Date	June 21, 2010
Author(s)	TUWIEN

Abstract

This document describes version 3 of the preservation planning tool *Plato*, thus it is the accompanying document to the software deliverable. It gives an overview of the history of the planning tool's development, explains how to install and access Plato, and gives an overview of events and case studies.

Keyword list

Digital Preservation, Preservation Planning, Tool support, Prototype, Distributed Preservation Services, Migration, Characterisation, Preservation Action, Registry, Design

Contributors

Person	Role	Partner	Contribution
Hannes Kulovits	Contributor	TUWIEN	
Christoph Becker	Main author	TUWIEN	
Michael Kraxner	Contributor	TUWIEN	
Stephan Strodl	Contributor	TUWIEN	
Mark Guttenbrunner	Contributor	TUWIEN	

Document Approval

Person	Role	Partner
Andreas Rauber	WPL	TUWIEN
Christen Hedegaard	Reviewer	KB-DK
Hans Hofman	SPL	NANETH

Distribution

Person	Role	Partner
Andreas Rauber	WPL	TUWIEN
Hans Hofman	SPL	NANETH
PP subproject mailing list		
Deliverables mailing list		
Christen Hedegaard	Reviewer	KB-DK

Revision History

Issue	Author	Date	Description
0.1	All	May 31, 2010	Alpha release
0.5	All	June 21, 2010	Release ready for review
1.0	All	June 30, 2010	Final version 3.0 released

EXECUTIVE SUMMARY

This document describes the deliverable PP/4-D5 Planets Planning Tool (version 3). Plato 3 is the third version of the preservation planning tool within the work package PP4 of Planets. This document provides an installation guide, an introductory user guide for the software, and an overview of case studies being conducted.

By first week of July 2010 Plato will be available from SourceForge: http://planets-suite.svn.sourceforge.net/viewvc/planets-suite/trunk/webapps/plato

A deployed version of the tool can be accessed at: http://www.ifs.tuwien.ac.at/dp/plato

TABLE OF CONTENTS

1. Introduction

To ensure digital content stays accessible and authentic for future users a plan has to be created taking into account legal and technical constraints such as storage space, infrastructure and delivery, copyright issues, and costs, user needs, and object characteristics.
A preservation plan defines a series of preservation actions to be taken by a responsible institution due to an identified risk for a given set of digital objects or records (called collection). The Preservation Plan takes into account the preservation policies, legal obligations, organisational and technical constraints, user requirements and preservation goals and describes the preservation context, the evaluated preservation strategies and the resulting decision for one strategy, including the reasoning for the decision. It also specifies a series of steps or actions (called ***preservation action plan****) along with responsibilities and rules and conditions for execution on the collection. Provided that the actions and their deployment as well as the technical environment allow it, this action plan is an executable workflow definition.*[1]
The four-phase high-level workflow shown below can further be divided into 14 steps. Evaluation of candidate actions uses controlled experiments and increasingly automated measurements.

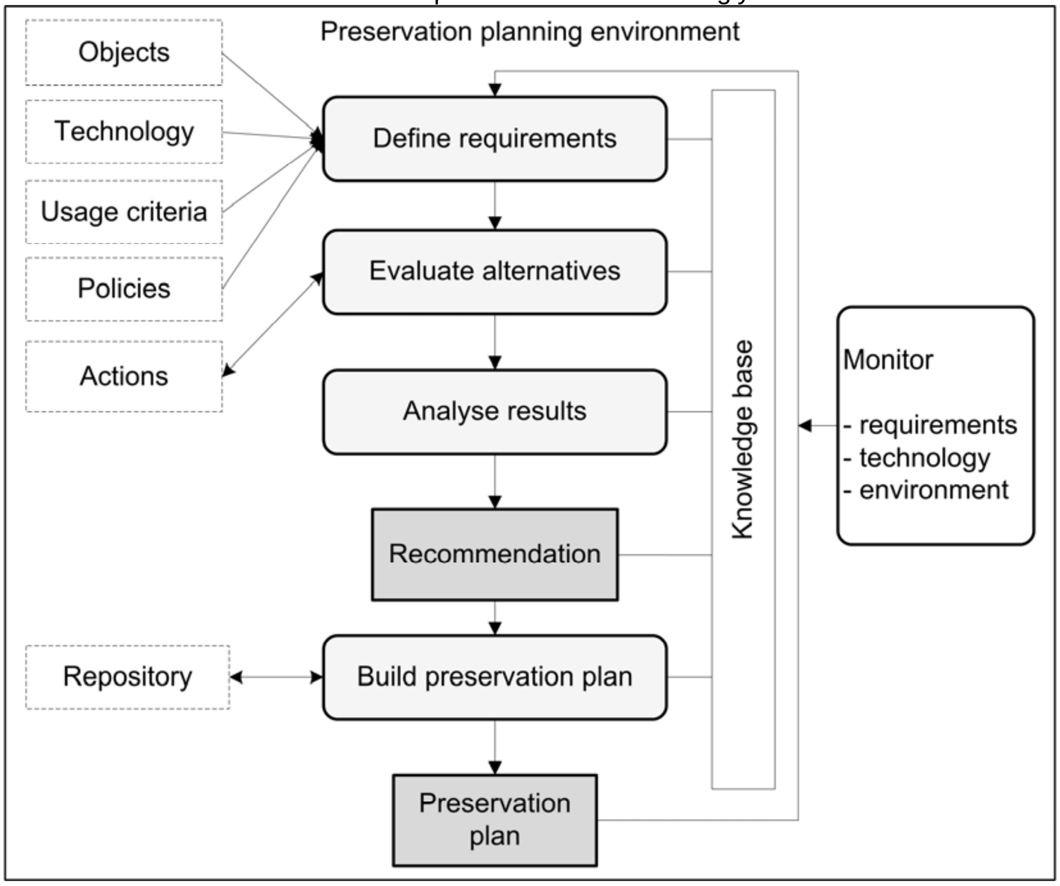


Figure 1 Preservation planning environment

Potential action tools are applied to sample content and evaluated according to a hierarchy of requirements, based on Utility Analysis. [3] A service-oriented framework greatly automates experiments and allows users to leverage various publicly available web service registries that provide access to potential preservation action tools. Quality-aware services measure execution parameters and quality of the action tools and take this burden off the experimenter. [5, 6]

The four phases of the main workflow are implemented in 14 steps, as described in detail in the user manual and a recent IJDL paper. [3] Figure 2 shows miniature screens of the resulting workflow steps.

[1] Full definition available at http://www.ifs.tuwien.ac.at/dp/plato/docs/plan-template.pdf

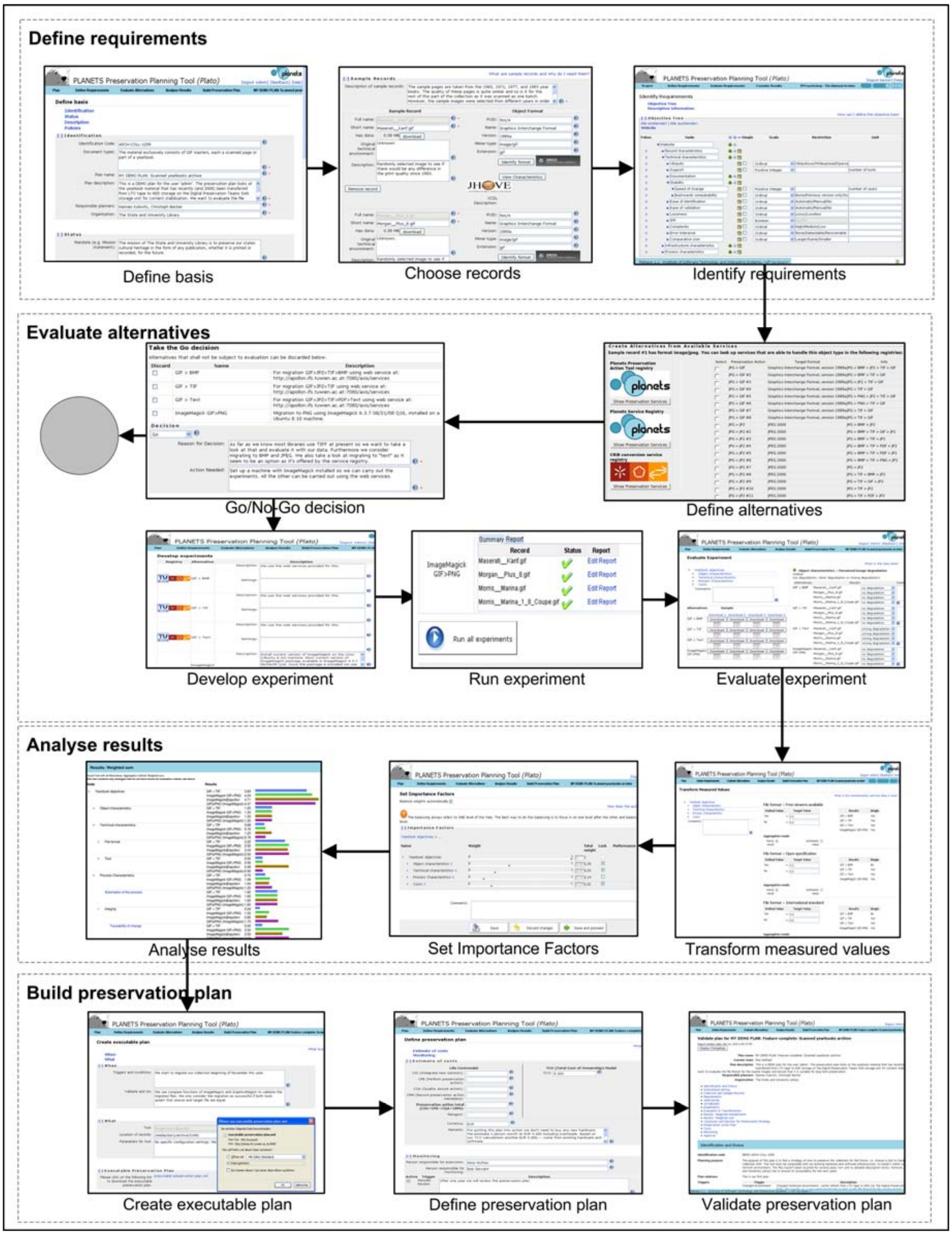
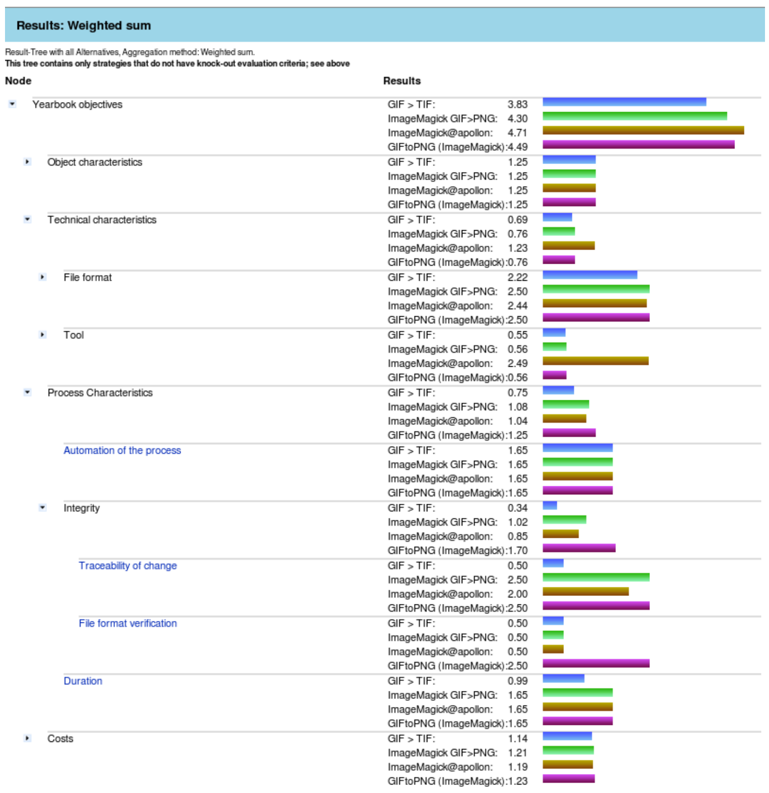


Figure 2 The planning workflow in Plato



Results: Weighted sum

Result-Tree with all Alternatives, Aggregation method: Weighted sum.
This tree contains only strategies that do not have knock-out evaluation criteria; see above

Node	Results	
Yearbook objectives	GIF > TIF:	3.83
	ImageMagick GIF>PNG:	4.30
	ImageMagick@apollon:	4.71
	GIFtoPNG (ImageMagick):	4.49
Object characteristics	GIF > TIF:	1.25
	ImageMagick GIF>PNG:	1.25
	ImageMagick@apollon:	1.25
	GIFtoPNG (ImageMagick):	1.25
Technical characteristics	GIF > TIF:	0.69
	ImageMagick GIF>PNG:	0.76
	ImageMagick@apollon:	1.23
	GIFtoPNG (ImageMagick):	0.76
File format	GIF > TIF:	2.22
	ImageMagick GIF>PNG:	2.50
	ImageMagick@apollon:	2.44
	GIFtoPNG (ImageMagick):	2.50
Tool	GIF > TIF:	0.55
	ImageMagick GIF>PNG:	0.56
	ImageMagick@apollon:	2.49
	GIFtoPNG (ImageMagick):	0.56
Process Characteristics	GIF > TIF:	0.75
	ImageMagick GIF>PNG:	1.08
	ImageMagick@apollon:	1.04
	GIFtoPNG (ImageMagick):	1.25
Automation of the process	GIF > TIF:	1.65
	ImageMagick GIF>PNG:	1.65
	ImageMagick@apollon:	1.65
	GIFtoPNG (ImageMagick):	1.65
Integrity	GIF > TIF:	0.34
	ImageMagick GIF>PNG:	1.02
	ImageMagick@apollon:	0.85
	GIFtoPNG (ImageMagick):	1.70
Traceability of change	GIF > TIF:	0.50
	ImageMagick GIF>PNG:	2.50
	ImageMagick@apollon:	2.00
	GIFtoPNG (ImageMagick):	2.50
File format verification	GIF > TIF:	0.50
	ImageMagick GIF>PNG:	0.50
	ImageMagick@apollon:	0.50
	GIFtoPNG (ImageMagick):	2.50
Duration	GIF > TIF:	0.99
	ImageMagick GIF>PNG:	1.65
	ImageMagick@apollon:	1.65
	GIFtoPNG (ImageMagick):	1.65
Costs	GIF > TIF:	1.14
	ImageMagick GIF>PNG:	1.21
	ImageMagick@apollon:	1.19
	GIFtoPNG (ImageMagick):	1.23

Figure 3 Visualisation of results

The result of using Plato is a complete preservation plan that can be deployed and executed.

A deployed version of Plato can be found at http://www.ifs.tuwien.ac.at/dp/plato .

The main new features of Plato 3 are:

- **Automated measurements**. We have developed an extensible measurement framework to connect decision criteria to measurable properties and metrics in six categories:
  1. Object outcome: Desired properties of the object such as editability, and properties of the original that need to be preserved, such as image width.
  2. Outcome format: Criteria on the format the objects will be stored in, such as standardization.
  3. Outcome effects: Effects induced by the preservation action, such as costs.
  4. Action Runtime: Runtime properties such as used time and memory.
  5. Action static: Action properties such as licensing costs and quality of documentation of a specific tool.
  6. Action Judgement: Properties that need to be judged, such as usability.
- **Fast-track evaluation**. We have added a fast evaluation workflow that leads from a few basic assumptions to a quick evaluation of potential solutions in just three steps.
- **Remote emulation**. The remote emulation service powered by GRATE is now integrated in PLATO and available through the Planets service registry. Please note that this is running on a different server than Plato and be patient.
- **Improved knowledge base**. The knowledge base for criteria and templates has been improved and its usage eased.
- **Integrated FITS**, the File Information ToolSet. FITS (http://code.google.com/p/fits) not only includes and homogenizes the output of characterization tools such as DROID (https://sourceforge.net/projects/droid) and JHove (http://hul.harvard.edu/jhove), but also includes additional extractors such as the ExifTool (http://www.sno.phy.queensu.ca/~phil/exiftool) and others. It is used to describe sample objects and support the automated evaluation of experiments.

- **Integrated P2**, the Semantic web registry. P2 is a format registry that is semantically enhanced and contains information to specifically support the process of digital preservation. To increase the number of alternative actions offered and allow automated format evaluation through the measurement framework, we have included 44.000 facts about formats as they are described in the semantic web registry P2. This also enables querying tools available that are able to convert a certain format but are not contained in the service registries since they have not been wrapped.
- **Ready for ePrints repository**. Plato can now construct preservation plans that are understood by ePrints (http://www.eprints.org). The repository acts upon a preservation plan created in Plato, i.e. automatically carry out the recommended preservation action.
- There is a number of small improvements scattered across the tool, often addressing feedback we received from the user community. One example is the new option of downloading the requirements tree to edit it on the client using the mind mapping tool Freemind. You can thus do round-trip editing using Freemind and the planning tool.

The main new features of Plato 2 were:

- **Extended workflow for plan definition.** The fourth phase of the Planets PP workflow, where a **preservation plan** is defined, has been implemented.
- **File format identification.** Plato now facilitates automatic file format identification with DROID (Digital Record Object Identification).
- **XCL characterisation and automated object comparison.** We have integrated the XCL engine and provide its functionality through the Plato application. It is furthermore possible to compare specific properties automatically between original and transformed objects using the Planets XCDL comparison service.
- **Template and fragments library.** A library of reusable fragments and templates assists the objective tree definition and reuse of recurring requirements such as significant properties of objects.
- **Service Integration** First migration services have been integrated.
- **XML import and export.** You can export completed projects to XML files, and import them again.
- **Detailed preservation plan.** The preservation plan has been extended. Parts of the enhancements made to the preservation plan were motivated by TRAC (Trustworthy Repositories Audit and Certification Checklist).
- **Online help.** For the more complex steps, we have written explanatory help pages describing how to use the tool.

2. Related publications

The development of Plato and its application to real-world problems has led to a number of additional developments, analyses and insights. The following publications provide a deeper insight into the corresponding outcomes.

[1] Christoph Becker and Andreas Rauber.
Improving component selection and monitoring with controlled experimentation and automated measurements.
Information and Software Technology, Volume 52, Issue 6, June 2010, Pages 641-655.

[2] Christoph Becker, Hannes Kulovits, and Andreas Rauber.
Trustworthy Preservation Planning with Plato.
ERCIM News 80, p.24-25, January 2010.

[3] Christoph Becker, Hannes Kulovits, Mark Guttenbrunner, Stephan Strodl, Andreas Rauber, and Hans Hofman.
Systematic planning for digital preservation: Evaluating potential strategies and building preservation plans.
International Journal on Digital Libraries (IJDL), December 2009.

[4] Hannes Kulovits, Andreas Rauber, Anna Kugler, Markus Brantl, Tobias Beinert, Astrid Schoger.

From TIFF to JPEG 2000? Preservation Planning at the Bavarian State Library Using a Collection of Digitized 16th Century Printings
D-Lib Magazine November/December 2009
http://www.dlib.org/dlib/november09/kulovits/11kulovits.html

[5] Christoph Becker, Hannes Kulovits, Michael Kraxner, Riccardo Gottardi, Andreas Rauber, and Randolph Welte.
Adding quality-awareness to evaluate migration web-services and remote emulation for digital preservation.
In: Proceedings of the 13th European Conference on Digital Libraries, ECDL 2009, Corfu, Greece, September 2009. LNCS 5714, Springer.

[6] Christoph Becker, Hannes Kulovits, Michael Kraxner, Riccardo Gottardi, and Andreas Rauber.
An Extensible Monitoring Framework for Measuring and Evaluating Tool Performance in a Service-oriented Architecture.
In: Proceedings of the 9th International Conference on Web Engineering (ICWE 2009), San Sebastian, Spain, June 2009. LNCS 5648, Springer.

3. Installation guide

This chapter describes the installation of the PLANETS Planning Tool (Plato).

This only applies if you want to run and deploy your own Plato server. For using Plato to create your own preservation plans, a Plato server is available at the Vienna University of Technology at the URL http://www.ifs.tuwien.ac.at/dp/plato

For information on how to use Plato, please refer to the user manual which is available on the Plato homepage.

The installation comprises three steps:

1. Download source code
2. Package generation
3. Deployment

The installation steps are described below.

Prerequisites

Before Plato can be installed (i.e. deployed) the following requirements must be met:

1. Java Development Kit (JDK)
   In order to install Plato the Java Standard Edition Development Kit (JDK) 1.5.0 must be installed on the target machine. (Due to a known issue in the underlying Interoperability Framework with the used web services stack, Java 6 cannot be supported at the moment. This shall be resolved in the next version.) Java can be installed following the instructions at: http://java.sun.com/javase/downloads/index_jdk5.jsp
2. Apache Ants
   The installation scripts are written for Apache Ant. It can be download at: http://ant.apache.org/bindownload.cgi

3. Planets Interoperability Framework
   Plato requires the IF for integration. This must be downloaded from the Planets GForge and installed following the corresponding instructions.

4. Eclipse IDE with JBoss IDE for Eclipse [optional]
   Plato was developed with the Eclipse SDK. Plato can be checked out as an Eclipse project.
   Eclipse is available at: http://www.eclipse.org/downloads
   JBoss IDE Eclipse Plug-in is available at: http://labs.jboss.com/jbosside

Download source code

By the first week of July 2010 the source code can be checked out from the PLANETS Planning Tool source code repository which is available at http://planets-suite.svn.sourceforge.net/viewvc/planets-suite/trunk/webapps/plato. The repository is organised as follows:

- **trunk** contains the latest source code in the main development stream
- **branches** contains branches that may be developed independently and in parallel to trunk
- **tags** contains snapshots of the source code repository at certain points/dates

Package generation

In order to deploy Plato on the server two property files must be adapted:

1. *build.properties* defines the location of the Interoperability Framework web server. The variable jboss.home must point to the IF web server (e.g. d:\bin\jboss, /home/username/jboss)
2. *server.properties* configures the host name of the web server. By default it is set to localhost.

After the two files have been adapted, Plato can be built and deployed. This is done by executing two ANT targets provided by build.xml.

1. buildall
2. deploy

When deploying Plato for the first time the data source file (plato-derby-ds.xml) must be deployed. This is done by running the ANT file build.xml with the target deploy-datasource.

Please note that Plato has been developed as an Eclipse project and can therefore also be comfortably built and deployed with the help of the Eclipse IDE.

4. Events

Plato has been presented at a series of international events, ranging from training courses organised by Planets, DPE, wePreserve, JISC, KeepIt, and Nestor, to conference tutorials at ECDL, JCDL, ICADL and RCDL to invited presentations around the globe.

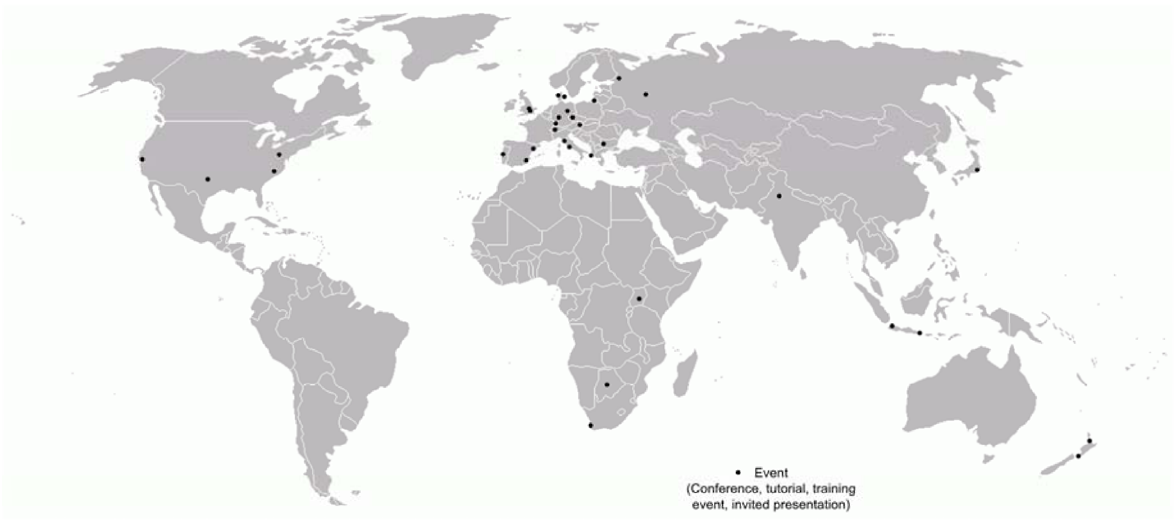

• Event
(Conference, tutorial, training event, invited presentation)

Figure 4 Plato events

An up-to-date list of events is available at the Plato homepage.

5. User accounts

Since the first release, a steadily increasing number of user accounts have been created. The development of user accounts since the public reference deployment was published is shown in Figure 5. In June 2009, the planning tool had 241 users who had created 135 preservation planning projects. By January 2010, there were over 430 users from Europe, North America, Russia, Asia, and Oceania, and the number of plans had almost doubled. Currently, there are over 600 user accounts registered from 46 different top level domains, and the entry page consistently comes up number 1 on a google query "preservation planning". Figure 6 shows the distribution of user accounts. The strongest groups are from the UK, Germany, Austria, The Netherlands, and US-American universities (.edu).

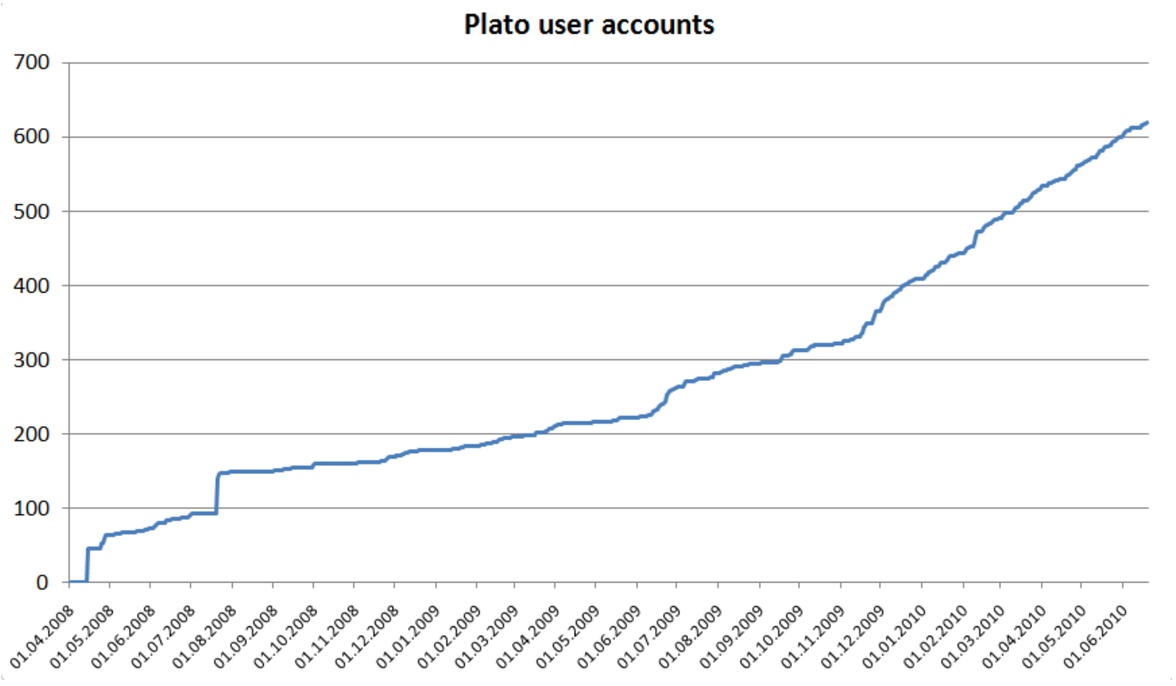


Figure 5 User accounts created since Plato 1.0

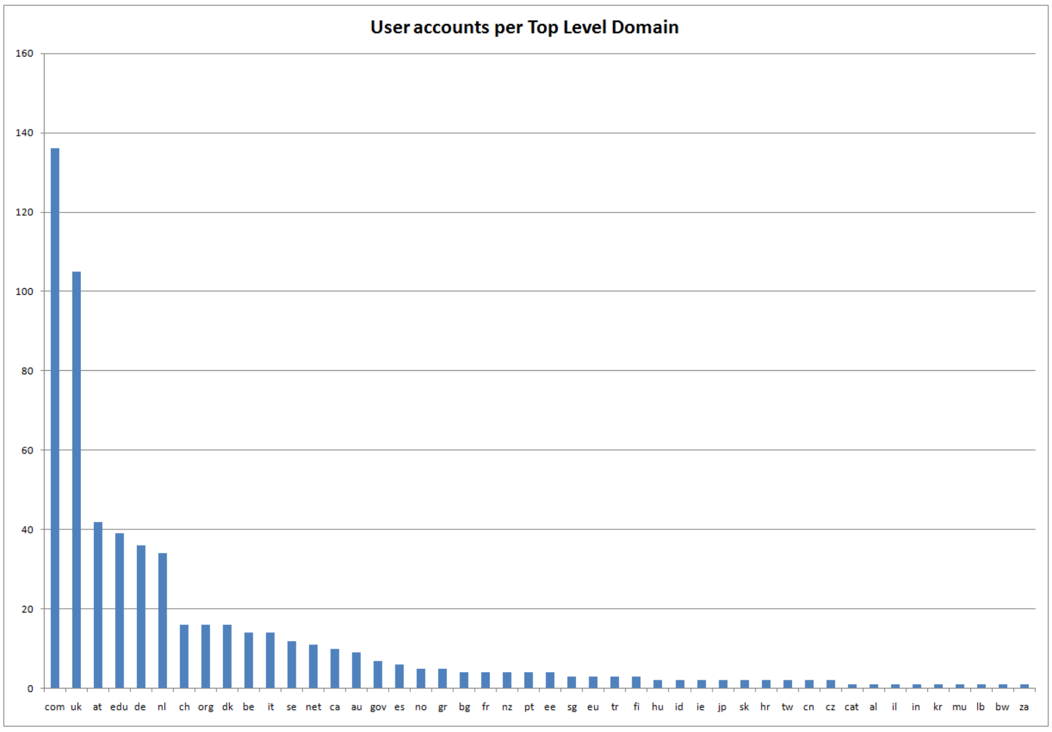


Figure 6 Distribution of user accounts

6. Architecture

Plato is based on a number of standard open-source technologies for the Java Enterprise platform. Figure 7 illustrates the layered architecture and summarises the core technologies and libraries involved. The components Workflow engine, User management, Logging, Service registry and Migration wrappers are provided by the Interoperability Framework.

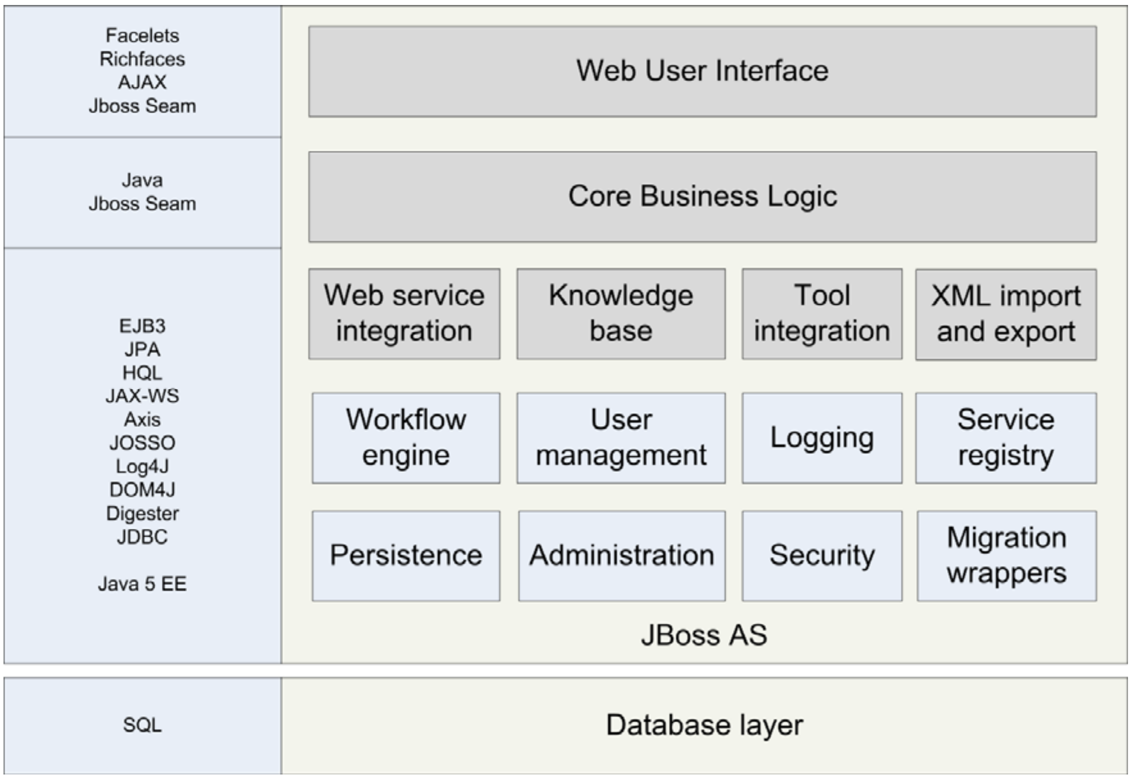


Figure 7 Layered architecture of Plato

The increasing degree of automation in Plato leverages integration of a variety of tools and information sources. Figure 8 summarises the essential aspects of these. The planning core is connected to a knowledge base containing measurable criteria, tree fragments and requirements templates. A component integration layer connects to the Planets service registry, the CRIB migration services, GRATE, and the MiniMEE engine included in Plato. Characterisation and registry adaptors include the XCL toolsuite, common tools such as DROID, JHove and FITS, but also access to the RDF statements from the semantic web registry P2 to support the automated evaluation of formats.
The repository adaption layer on the right is in dotted lines as it is in its infancy. We are working on integrating Plato with a number of repositories; ePrints in its newest release is the first repository to be able to act upon plans created by Plato and execute the defined actions directly.

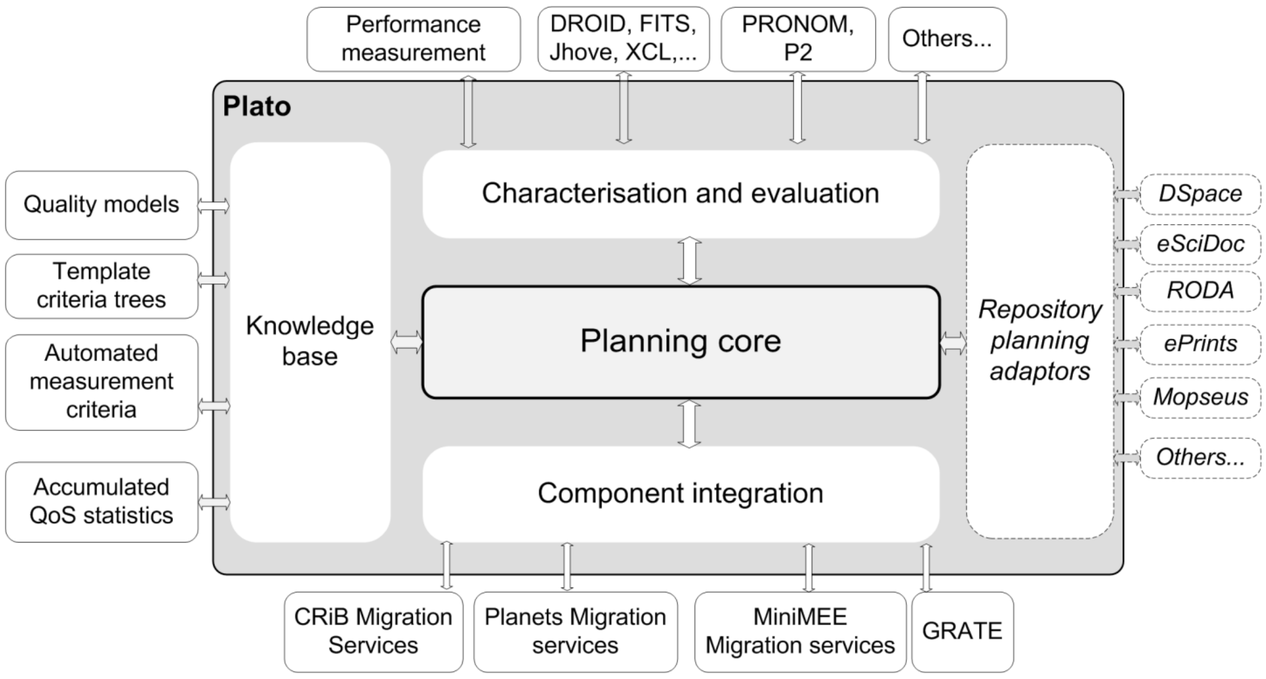


Figure 8 Integration architecture