Specification for the Open-PEOPLE software platform (user side)

Sophie Alexandre, Jonathan Ponroy, Olivier Zendra

INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

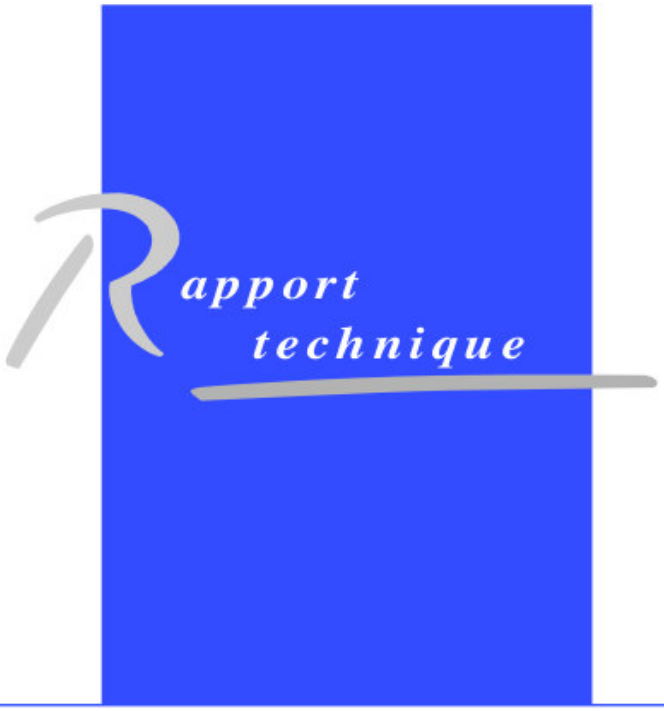
Specification for the Open-PEOPLE software platform (user side)

Sophie Alexandre, Jonathan Ponroy, Olivier Zendra

N° 0410

Mars 2011

Thème : Systèmes embarqués, temps réel



INSTITUT NATIONAL
DE RECHERCHE
EN INFORMATIQUE
ET EN AUTOMATIQUE

INRIA

centre de recherche NANCY - GRAND EST

Specification for the Open-PEOPLE software platform (user side)

Sophie Alexandre, Jonathan Ponroy[1], Olivier Zendra[2]

Thème : Système embarqué
EPI Trio
Projet ANR Open-PEOPLE

Rapport technique n 0410 – Juin 2011 - 46 pages

Abstract: Open-PEOPLE is a project which aims at estimating the energy consumption of modelled hardware and software architectures with several level of accuracy. This document presents the functional specification of the Open-People Software Platform (OPSWP), ranking functionalities by priority (*required, highly desired, nice to have*). It also contains use cases and activity diagrams pertaining to the OPSWP.

Keywords: Open-PEOPLE, software platform, specifications, requirements, functionalities, use case

[1] Inria Nancy Grand Est – jonathan.ponroy@inria.fr
[2] Inria Nancy Grand Est – olivier.zendra@sinria.fr

Centre de recherche INRIA Nancy - Grand Est
615, rue du Jardin Botanique
CS 20101
54603 Villers-lès-Nancy Cedex (France)
Téléphone : +33 03 83 59 30 00 — Télécopie : +33 03 83 27 83 19

Specification for the Open-PEOPLE software platform (user side)

Résumé: Le projet Open-PEOPLE est un projet visant à offrir la possibilité d'estimer la consommation d'architectures matérielles et logicielles modélisées à différents niveaux de précision. Ce document introduit les spécifications fonctionnelles par niveau de priorité (*obligatoire, souhaité, secondaire*). Il précise également les différents cas d'utilisation attendu par la plateforme logicielle et fournit différents diagrammes d'activité.
Mots clés: Open-PEOPLE, plateforme logicielle, spécifications, exigences, fonctionnalités, cas d'utilisation

INRIA

Table of Contents

1 Introduction

The Open-PEOPLE (Open Power and Energy Optimization PLatform and Estimator) project is financed since the end of 2008 by ANR (Agence Nationale de la Recherche), the French National Research Agency. Open-PEOPLE initially gathers 5 partners from academia and 2 from industry. This project aims at providing a federative and open platform for the estimation and optimization of power and energy consumption in computer systems. The platform users will be able to evaluate application consumption on a hardware architecture chosen among a set of provided typical, parametric architectures. In the considered system, the components will be picked from a library of hardware and software components, be they parametric or not. It will be possible to perform the estimation at various stages of the specification refinement, thanks to a methodology based on multi-level, interoperable and exchangeable consumption models allowing an easy exploration of the design space. Thus, estimations results may be used to check the energy behaviour of a system developed with simulation platforms. Feedback about the application functional properties will allow further refining of the estimation results in Open-PEOPLE. A standardisation of consumption models will be proposed in order to allow interoperability and have easier exchanges with other platforms. The Open-PEOPLE library of consumption models will be extendible: new component models will be added as the user applicative requirements evolve and as implementation techniques progress. To do so, the software estimation platform that will be accessible via an Internet portal shall be linked to a hardware platform made of an automated measurement testbench, which will be controllable from the software platform. A standalone version will also be provided to meet the confidentiality requirements of industry. A library of applications benchmarks will be proposed to characterize new components and new architectures. In addition to the research work required to build methods for multi-level estimation in heterogeneous complex systems, research work shall be carried on in order to offer new methods and techniques making it possible to optimize consumption thanks to the results provided by Open-PEOPLE.Open-PEOPLE is hence geared towards academia to support research work pertaining to consumption estimation and optimization methods, as well as towards industry to estimate or optimize the consumption of future products.

2 Preface

2.1 Table of versions

Version	Date	Description & rationale of modifications	Sections modified
0.1		First draft	26/02/10
1		Release	30/04/10

2.2 Table of references and applicable documents

Reference	Title & edition	Author or editor	Year
Open-PEOPLE_Document_B.pdf	**Open Power and Energy Optimization Platform and Estimator ANR Proposal**	Eric SENN	2008
D11PlatformFunctionalitiesv10.pdf	**Platform functionalities**	Dominique Blouin	2009

2.3 Acronyms and glossary

Term	Description
AADL	Architecture and Analysis Design Language
CAT	Consumption Analysis Toolbox
OSATE	Open Source AADL Tool Environment
PCMD	Power Consumption Models Development
PCAO	Power Consumption Analysis and Optimization
FPGA	Field Programmable Gate Array
MPSoC	Multiprocessor System-on-Chip
CLB	Configurable Logic Blocks
UML	Unified Modeling Language
MARTE	Modeling and Analysis of Real-Time Embedded systems (an UML Profile)
VHDL	VHSIC Hardware Description Language
VHSIC	Very High Speed Integrated Circuit
RCP	Rich Client Platform
OSGi	Open Services Gateway Initiative
SWT	Standard Widget Toolkit

3 Executive Summary

This document presents the specification of the software part (called "software platform") of our unified hardware-software Open-PEOPLE platform. This software platform is to provides energy usage estimations for benchmarks run on a chosen architecture.

This software platform is designed according to the users requirements, especially from a functional and accessibility point of view.

Its main intent is, as a testing infrastructure, to be the main tool upon which the research works about low-power and low-energy can be carried on. As such, it provides the framework into which the models and prototype tools produced by the modeling and estimation work will be integrated. Integration and polishing of the tools is part of the work pertaining to this platform.

The platform is to be accessible via a secure Internet portal as well as locally.
It should be an open-source tool, free to use and download for anyone.

4 Scope of the Document

The aim of this document is to present the Open-PEOPLE software platform specification, elaborated by taking into account the functional requirements that have emerged from users.
The document describes the software platform architecture and defines which functionalities are in the perimeter of this software platform.

This is not an implementation document: implementation choices are presented from a high level point of view; in addition, the user interfaces presented in the document may not be the definitive ones.

5 Description of the software platform

5.1 Overview

The Opn-PEOPLE software platform is part of a more global platform, which includes :

- the software platform core itself that will integrate energy models and energy estimation tools,
- all the required interfaces between the software and the underlying hardware platform and in particular a front end Internet portal,
- the hardware platform for actual measurements and models development,

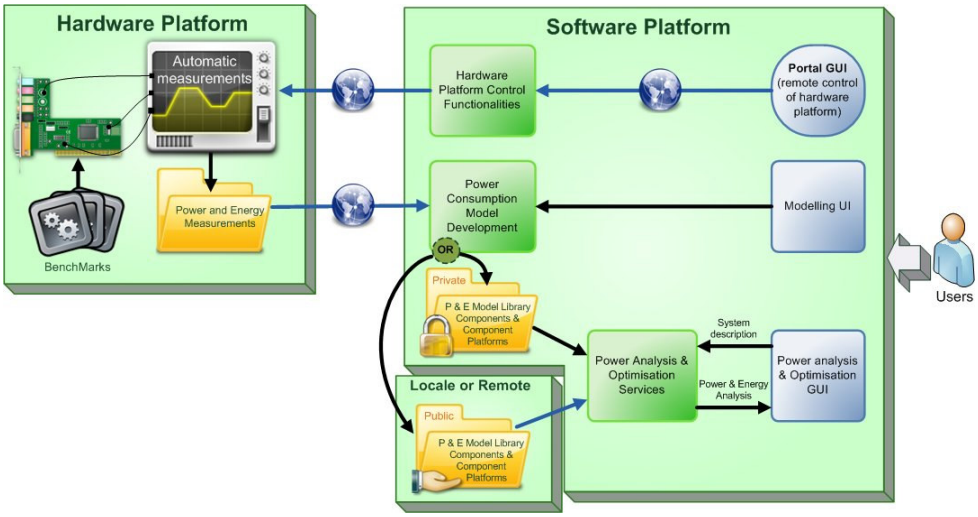


Figure 1 : The global Open-PEOPLE platform

5.2 Organisation of the software platform functionalities

The architecture of our software platform features functionalities that have emerged from the th users requirements.

They can be divided into two main categories:

- The Power Consumption Models Development (PCMD) functionalities: these functionalities are used for the development of the power consumption models and their validation on a hardware platform.

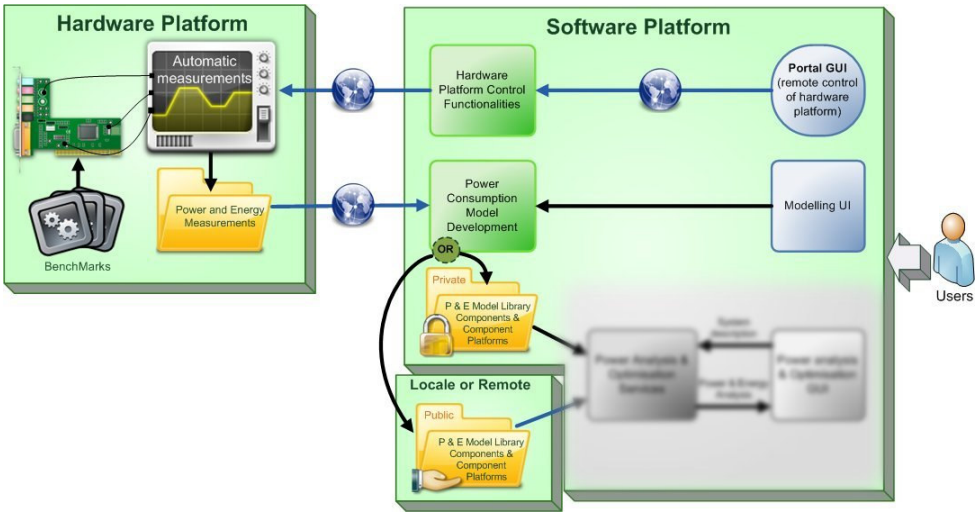


Figure 2 : The software platform: the PCMD functionalities

- The Power Consumption Analysis and Optimization (PCAO) functionalities : they allow an embedded system designer to perform the power consumption analysis and optimization on the designed system. They use the power estimation models defined during the consumption models development step.*

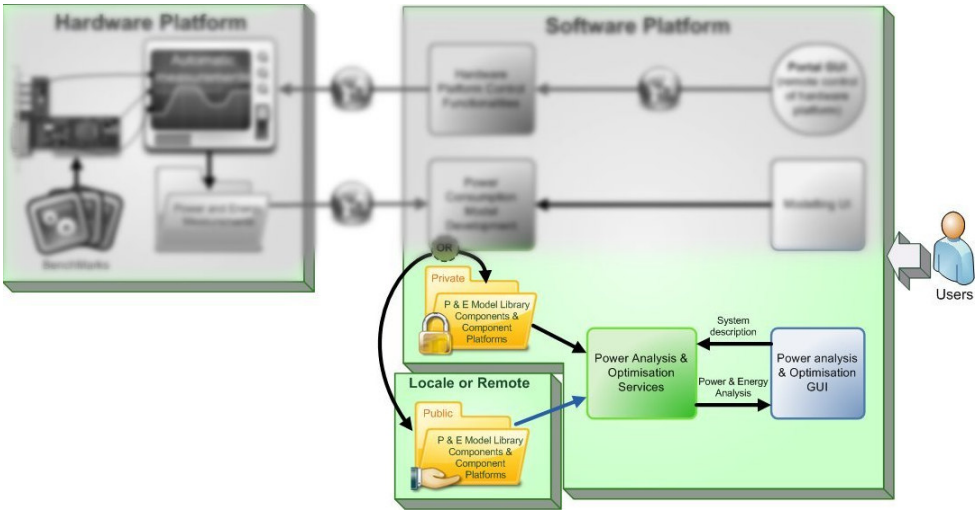


Figure 3 : The software platform: the PCAO functionalities

5.3 Architecture of the software platform

An important requirement for the software platform is to ensure the confidentiality of some of the data and source code belonging to the industrial partners. This means that the software platform must be able to be executed locally without Internet connection in order to never risk exposing the proprietary data.

On another hand, it might be useful for the partners (industrial or not) to exploit the software platform by accessing, via Internet, some data such as libraries of models stored on a remote

server. Indeed, some users who won't need as much confidentiality might prefer the advantages of cen-tralized information and not to have to store the data locally.
To this end, we will provide a software platform that can be exploited locally in an off-line mode or via a secure Internet access. The mode of use will depend from the user.
Another important feature to take into account for the choice of the platform architecture are the graphical interfaces. They must have graphical functionalities sufficiently "advanced" to be able to manipulate graphical representations of the hardware components, while not being too complex to use.
They must be able to offer different levels of use : for example, description of the system of components designed for analysis by a textual or a totally graphical way.
In this context, the graphical features needed (for example : Drag&Drop) exceed what can be offered by simple HTML web pages and make us turn towards rich interfaces.
The implementation choices for the software platform also depends on the available technical solutions, which should be mature and open source.
By taking into account these considerations, we propose to implement the software platform by an application having the following characteristics :

- it will be a thick client,
- it will be **a standalone application** stored locally on the user's desktops.

(it may also be hosted on a server on an Intranet.

- it will have **two possible modes of use** :
    - **the off-line mode** : in this mode, data, models or tools needed and used by the application will be stored locally with the standalone application on the user's computer.

(see Figure 4 : off-line mode),(possibly on a server on an Intranet, according to the usage of some partners).

    - **the Internet access mode** : in this mode, the data, models or tools needed and used by the application are stored on a remote server to which the standalone application will be accessed via an Internet connection.

(see Figure 5 : Internet access mode),

- its mode of use depends only from the user choice,
- its presentation layer will be designed with rich graphical user interfaces.

5.3.1 Off-line mode

- The standalone application will be installed and updated on the client side, that is to say locally on the user's desktops.
- The data, libraries, models or tools needed and used by the application will be stored locally with the standalone application on the user's computer. It will allow to store and use the confidential and proprietary data without risk to disclose them.

(Note that in the two previous points the user's desktop can be eventually replaced by server on an Intranet).

Standalone application (Thick client)

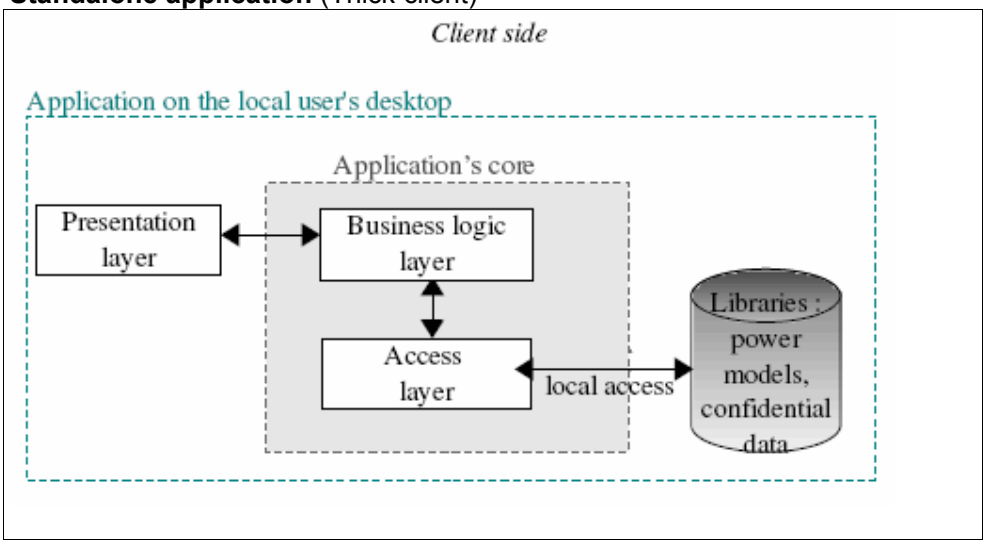


Figure 4 : off-line mode

Implementation :
We are planning to develop a rich client application by using the Eclipse Rich Client Platform (RCP). RCP development is based on the use of the Eclipse plug-ins Development Environment (to create OSGi plug-ins), on the Eclipse Standard Widget Toolkit API (SWT API), on JFace and on the Eclipse workbench .
A tool like RCP Developer (more precisely SWT Designer, plugged-in the Eclipse framework) will be used for the development of the presentation layer.
Eclipse (via the RCP Developer suite) alos provides tools to easily manage the updates of our application or to produce help documentation for the application.

5.3.2 Internet access mode

- Like in the previously presented mode, the rich client application will be installed and updated on the client side, that is to say locally on the user's desktops.
  (possibly on a server on an Intranet).
- Data, libraries, models or tools needed and used by the application are stored on a remote server accessed via a network connection by the standalone application.

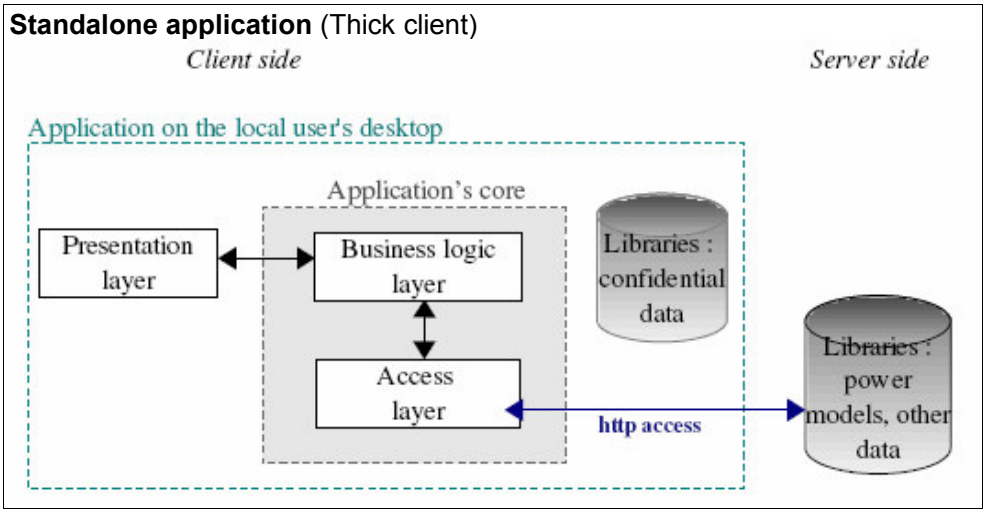


Figure 5 : Internet access mode

Implementation :
To implement the communication between the client and the server part, we plan to use a framework called ECF (Eclipse Communication Framework). This one allows the use of remote OSGi services on server side from the client side.

6 Generic features and functionalities of the software platform

The software platform will include a set of generic functionalities like : security, authentication to access the platform, authorization to access some parts of the platform, on-line printing, on-line help and the possibility to keep user preferences.
In the following, each listed feature and functionality is described in terms of requirements.
(they can be decomposed in one or several requirements).

6.1 General non functional features

(non functional requirements)

1) The software platform is designed in **a modular way** :

   - [REQNF-000] : It will be designed like a set of modules and more precisely with a plug-in architecture that will ensure the creation of a modular and extensible platform into which the estimation tools for power consumption analysis and optimization will be deployed.
     New modules and functionalities could be added by extending existing plug-ins, or by developing new ones. These may be based on the extension points offered by existing plug-ins and may use the results, services or functionalities provided by these last ones.

2) The developments will be **documented** :

   - [REQNF-005.1] : It will be important to document during the implementation step, the extension points provided by the plug-ins, and more generally, the interaction and dependencies between the plug-ins that compose the platform. (Internal technical documentation).

- [REQNF-005.2] : To help developers to extend or re-use plug-ins of the platform, a plug-in development guide can be provided. (A developer guide).

  Note about the development tool chosen :
  This plug-in development will be realized thanks to the OSGi technology (in Java) via the Eclipse platform, that we have already decided to use. (The Eclipse platform is itself constituted by plug-ins and use an implementation of the OSGi specification called Eclipse Equinox).

3) The software platform **licensing** :

- [REQNF-010.1] : It must be a completely open-source tool, free to use and download for anyone.
  We must make sure that any commercial licenses won't be necessary in any modules or tools provided with the software platform.
- [REQNF-010.2] : The software platform license must not impose restrictions on its use by end-users and must allow to use its code as well as a part of software released under a proprietary license.
  A permissive free software license will be used (such as BSD license, MIT license or EPL license).

4) The **ergonomics** :

The software platform will have quality and user-friendly graphical user interfaces, that will be intuitive to use.

- [REQNF-015.1] : The interfaces will have to allow to manipulate graphical representation of hardware components for easily designing the systems to analyze.
- [REQNF-015.2] : To ensure consistent look-and-feel among the interfaces, we could refer to a graphic charter.

5) The **confidentiality** :

Ensuring confidentiality of user data (or eventually source code) is an important requirement for the software platform especially for industrial partners.

- [REQNF-020] : The software platform will have the possibility to be executed locally without Internet connection in order to never disclose the proprietary data.
  (All data, libraries and tools required for an off-line mode being stored on the same computer than the application constituting the software platform).

6) The remote access :

- [REQNF-025.1] : The software platform will have the possibility to use a network connection in order to perform the update of the application itself
- [REQNF-025.2] : The software platform will have the possibility to use a network connection in order to upload or download models from libraries (such as libraries of power consumption models).

Required generic functionalities
(functional requirements)

1) <u>Configuration and change from off-line mode to Internet access mode :</u>

- [REQ-GNFCT-000] : The standalone application will propose to the user a configuration option concerning the access mode to the data needed for its work.

1. It can be done just after installation of the standalone application :

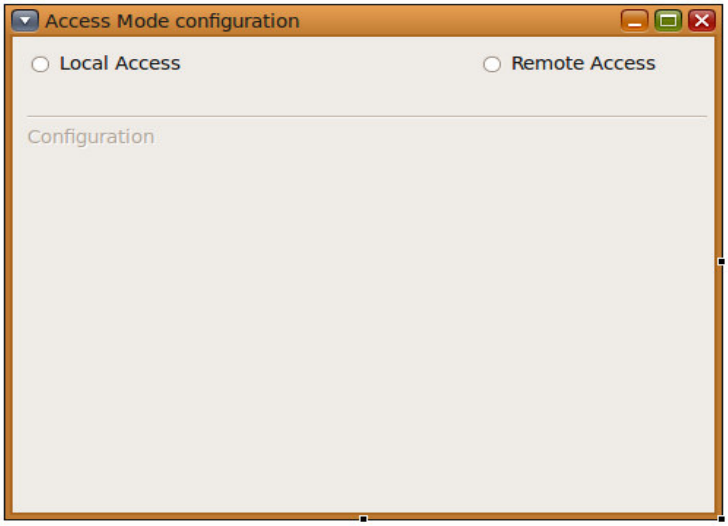


Figure 6 : Access Mode Configuration

- the user indicates that he wants to use a local access and he configures the access and paths to his private data.

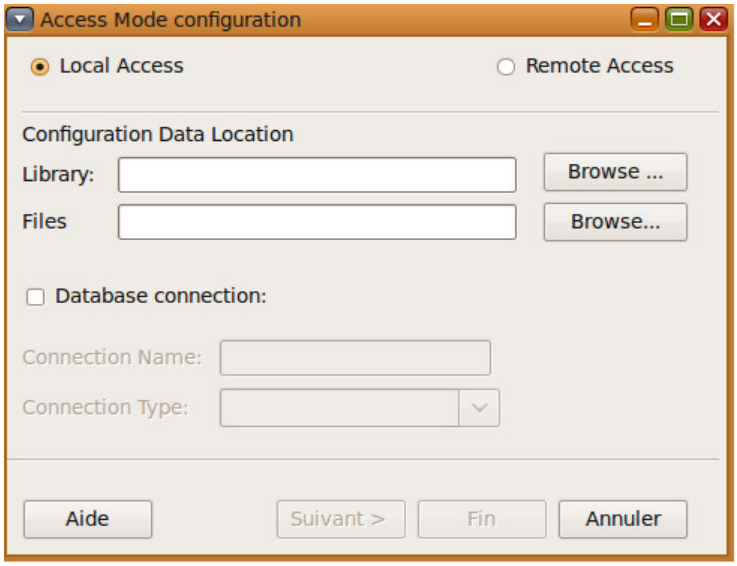


Figure 7 : Local Access configuration

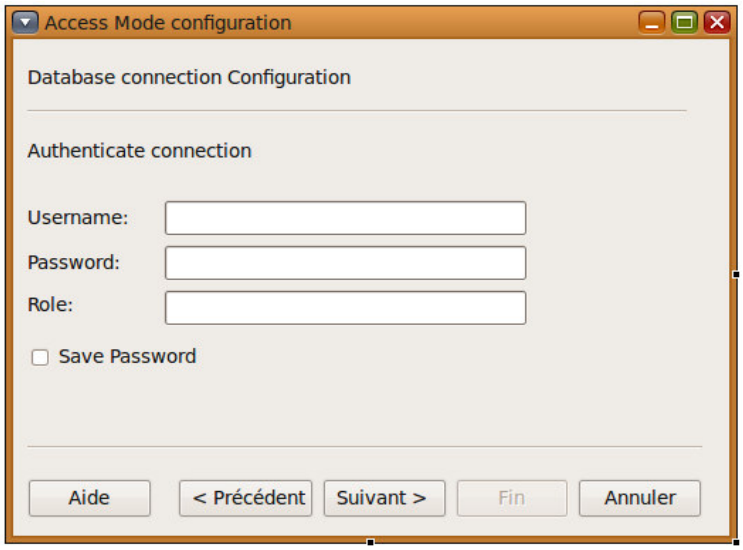


Figure 8 : Local access configuration : database connection

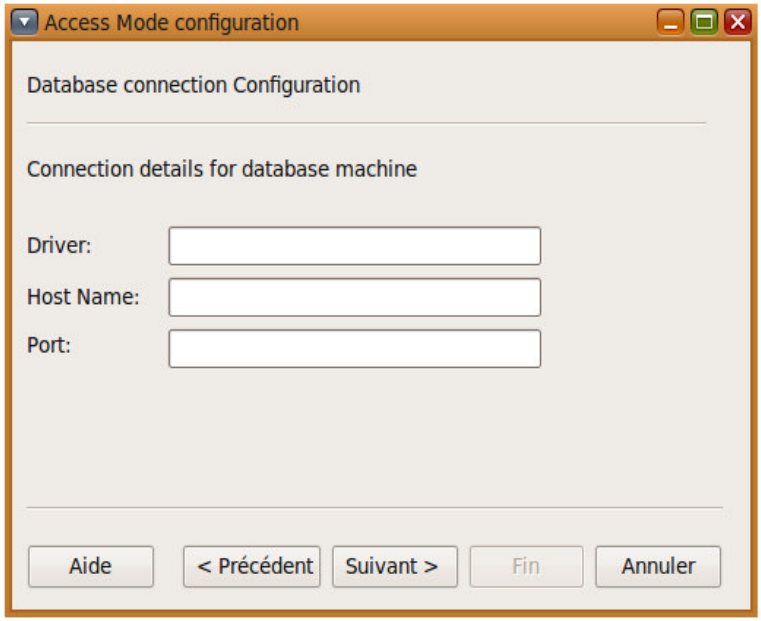


Figure 9 : Local Access configuration : database connection

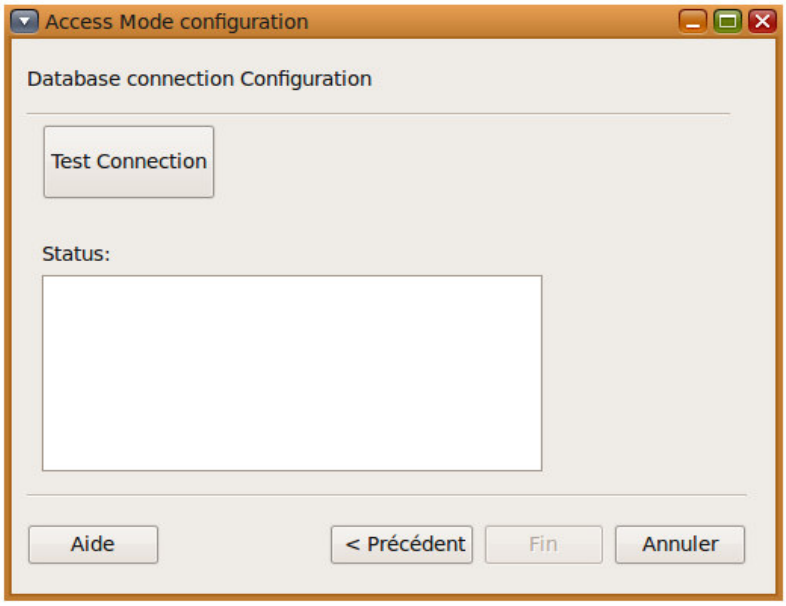


Figure 10 : Local Access configuration : database connection

- or in the opposite, the user chooses a network connection to some server.

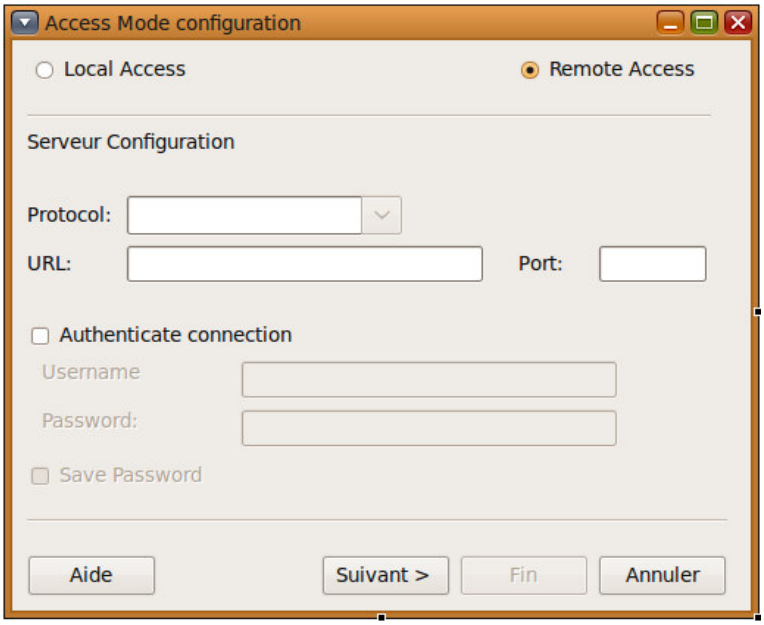


Figure 11 : Remote access configuration

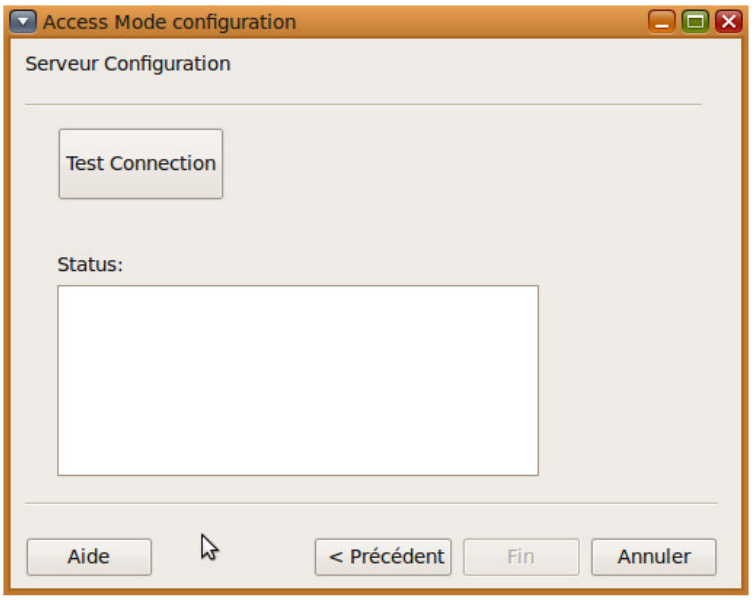


Figure 12 : Remote Access configuration

2. or it can be done later. While the user has chosen to work with the application in one of the modes, he could have access to a configuration menu to change the access mode.

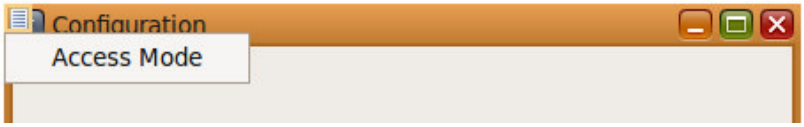


Figure 13 : Configuration Menu

the menu allows to open an "Access Mode configuration" window (Figure 6).
the user can choose to configure a local access (Figure 7).
the user can choose to configure a remote access (Figure 11).

2) <u>Software updates :</u>

- [REQ-GNFCT-005.1] : We will set up an updater to manage the upgrade of the software platform.
  (Note :
  - The installer package must be easy to create.
  - The developers could choose which plug-ins they will include in the build.
  - The installer package must be easily usable by the users.
  - The users could choose updating whole or part of the application thanks to the plug-in development.)
- [REQ-GNFCT-005.2] : We will provide to the end-user an installer package containing the RCP application files.
- [REQ-GNFCT-005.3] : We want to provide equally with the main application a set of tools (modeling tools, simulation tools, compilers, ...). We will choose current, well-known and open source tools. It will be a common and minimal base of tools to begin to work with the Open-PEOPLE platform.

<u>3) Authentication system</u>

- [REQ-GNFCT-010] : For any remote access by the software platform, authentication will be required to ensure security.
  (For example, to access the remote hardware platform during the model development step or to access libraries stored on a remote server).

4) Data encryption :

- [REQ-GNFCT-015] : To ensure confidentiality an encryption mechanism will be used for all data sent to remote server or application tier.
  So secured and encrypted access with SSL protocol will be always preferred to simple HTTP access.

5) Authorization :

- [REQ-GNFCT-020] : It will be necessary to implement a mechanism of authorization to control the access to some parts or functionalities of the software platform. For that a Role Based Authorization mechanism could be used.

We plan to have this three types of users with the following roles :

- admin :
  add accounts for new users.
- provider :
  realize measurements on a software platform,
  develop consumption models,
  validate consumption models.
- user :
  realize power and energy consumption analysis and optimization,
  access to the consumption models (validated or not).

6**) User documentation :**

- [REQ-GNFCT-025] : The software platform will provide to users all the needed documentation to each version : user manuals, possibly tutorials if needed.

6.2 Highly desired functionzlity

(functional requirement)

1) On-line help :

- [REQ-HDFCT-000] : The software platform could provide an on-line help easily accessible at any point of the software platform.
  The users could benefit from a complete help system providing features such as search in content, organized topics, links, internationalization, convenient interfaces and navigation in help.

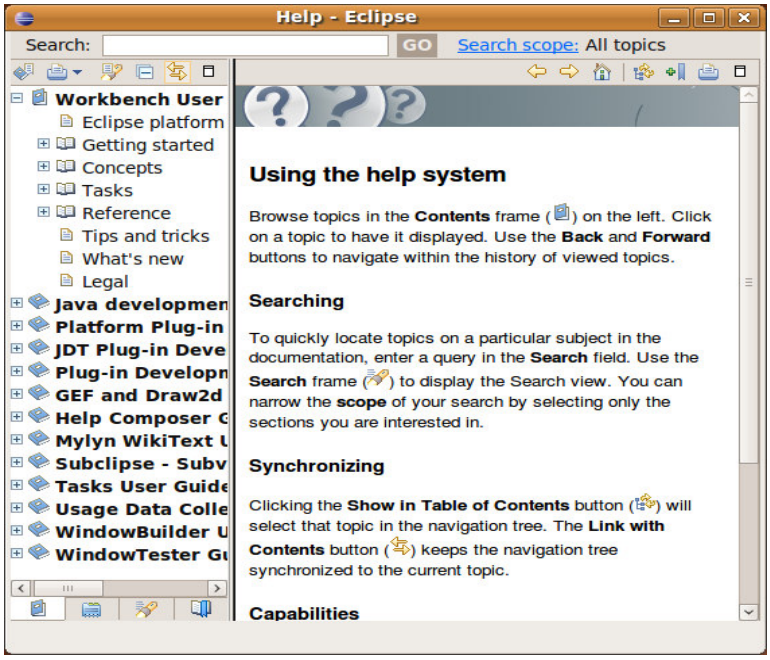


Figure 14 : Look of the online help system

6.3 Additional "Nice To Have" functionalities

1) <u>On-line printing :</u>

- [REQ-NTFCT-000] : The software platform could include a module to support the on-line printing of the data presented in the graphical interfaces.
  We can plan to propose direct printing or file printing.

2) <u>Reporting :</u>

- [REQ-NTFCT-005] : The software platform could include a reporting system : the aim being that a user can obtain on demand statistics and charts reports with the results (like power, energy, consumption profiles, …) of the power and energy consumption analysis.

3) <u>User preferences :</u>

- [REQNTFCT-010] : The software platform could include preference pages to store specific and customize user settings.
  The user could so find his preferences each time he accesses the software platform.

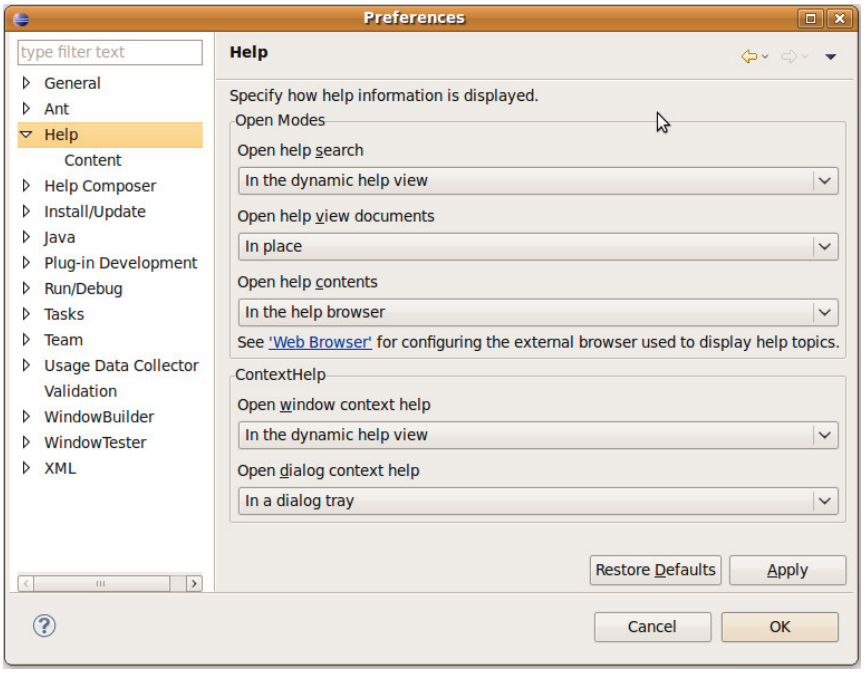


Figure 15 : Look&feel of a preference panel

7 Power Consumption Analysis and Optimisation

7.1 Description of system architecture

The functionalities are presented by a short description, also in term of requirements.
(they can be decomposed in one or several requirements).

7.1.1 Required functionalities

At first for designing the system architecture, it is necessary to agree about which Description Languages will be accepted in entry of the software platform.

1) The embedded system **description languages** :

To design embedded system made of components we must chose standardized description languages, understandable by the analysis tools and can be used as input languages of the software platform.
In fact, it is important to choose **a language of reference**. After talks with the different Open-PEOPLE partners, the high level abstraction description language chosen is : **AADL** (Architecture and Analysis Design Language).

Indeed it is already used by the different partners and especially the UBS ("Université de Bretagne Sud") has already developed a tool for power and energy consumption analysis : CAT (Consump-tion Analysis Toolbox) supporting the AADL language. (A tool that is planed to be part of the software platform).

For hardware descriptions, the following languages currently used by the partners have been retained : AADL, C, SystemC (C++), VHDL (for the FPGA), UML (MARTE : UML Profile for Modeling and Analysis of Real-time and Embedded Systems).

Note : The translations from the different languages to AADL will be provided by the different partners.

Requirements :

- [REQSYST-000.1] : The platform will use the reference language AADL as input.
- [REQSYST-000.2] : It will support other description languages (C, SystemC (C++), VHDL, UML (MARTE)) provided translations between them and the reference language AADL are available.

2) Extensibility of the system description language :

An extensible DSL (Domain Specific Language) will have to be available for allowing the description of new components.
An estimation model being most of the time only valid for a specific component, it is necessary to extend the DSL with the description of new components when power estimation models become available for these components.

Actually such a DSL, the PADL (Power Analysis Description Language) has been already developed by the UBS for use in the CAT estimation tool.
The PADL can be used in conjunction with a modeling language such as AADL or MARTE.
Automatic model transformation from AADL to the PADL is available.

The PADL is defined in a working document "power analysis description language" (by D.Blouin, E.Senn, UBS).
(This document can be consulted on the site http://open.people.fr; Intranet/Working documents).
Requirement :

- [REQSYST-005] : An extensible DSL (the PADL developed by the UBS) will be available in the software platform.

3) Refinement levels :

The description language used, will have to allow that the refinement of the architecture model be possible at components level.
For these ones, configuration parameters could be defined (such as operating frequency).
So the description language (multi-level) should allow to describe functionalities at different levels (some very precisely and others very briefly).

Requirement :

- [REQSYST-010] : The description language in the software platform will be a multi-level description language allowing the different refinement levels.

4) FPGA modeling :
It should be possible to include FPGA modeling in the system description.

FPGA consumption models will be developed and included in the Open-People model library.
The models will be built from hardware platform measurements.
The models will be defined in AADL and they should be transformed in the Domain Specific Lan-guage (PADL).

Several levels of description will be proposed in order to extract different types of consumption information.
A first level of description can be a black box model, with few information like number of CLB used for a specific task.
Other levels of description may be developed to extract more accurate information about the activity of each block included in a task, the number of memory accesses, etc.

The dynamic reconfiguration feature of the new FPGA circuits will be modeled in the DSL, with some adaptation if necessary.
According to this feature, the impact on the operating system will be studied, and behavioural model of execution will be developed.

As implementation of tasks is generally done with the VHDL language, the task descriptions may be associated to the corresponding AADL models.

Requirements :

- [REQSYST-015.1] : The software platform will allow the development of FPGA consumption models : they will be defined in AADL and should be transformed in PADL. The software platform will allow the use of several levels of description to extract different types of consumption information.
- [REQSYST-015.2] : The dynamic reconfiguration feature of the new FPGA circuits will be modeled in the DSL (PADL), with some adaptation if necessary.
- Behavioural model of execution will be developed from the study of the impact on the operating system.
- [REQSYST-015.3] : The software platform will allow that the descriptions of the tasks, which implementation is generally done with the VHDL language, may be associated to the corresponding AADL models.

5) Operating system (OS) modeling :

It should be possible for a user to model the OS in the system description.
It should be possible to provide advanced models of the power management on the operating system and to take into account an operating system with or without DVFS (dynamic voltage scaling and dynamic frequency scaling).

Requirement :

- [REQSYST-020] : The software platform will allow the modeling of operating system with or without DVFS and allow to provide advanced models of the power management.

6) Allocation strategies :

It will be interesting that the software platform allows users to test the allocation of software components on different hardware resources.
The user could choose to try different deployment model from the same system description
(for example, binding a thread on different processors, mapping data on several memories, ...)
The provided estimation tools (such as CAT) could take in charge the management of these successive deployment of software on the hardware resources.

The aim of this functionality is to give the users a way to improve the system design by observation and comparison of the performances obtained after analysis.

Requirement :

- [REQSYST-025] : The estimation provided tools will have to allow users to try different deployment models from the same system description.

7.1.2 Highly desired functionalities

1) Hardware interfaces description :

The software could allow users to make some control when they define the components system on which an estimation will be computed.
Indeed to avoid wrong architectures definitions, specific interfaces (like DDR-X or GMII) will have to be clearly defined in models. Then these interfaces will be used to validate the architecture.

(An example is the incompatibility between interfaces of a PQII and a DDR2 component which it will not be possible to connect).

The way to define and attach these interfaces description to the functional model is currently studied by UBS.
(A reference to a document describing this mean could be added later and can be certainly consulted on the site http://open.people.fr; Intranet).

Requirement :

- [REQHDSYST-000] : Hardware interfaces description will be available and can be attached to the functional model in order to provide some control during the components system definition and to avoid wrong architectures definitions.

2) User-defined models :

The software platform could offer this functionality for users who want to quickly create basic models in the case where the functional and the consumption models don't exist.
For example, the functional description (in AADL) would consist in defining a few working modes.
The consumption model would be a simple consumption value according to frequency, voltage values, ….
A "white box" of basic functional models defined by the user or consumption models could be attached to components of the system description.

Requirements :

- [REQHDSYST-010.1] : basic functional models defined by the user can be attached to components of the system description.
  This can be supported by the description language.
  (Possibly a graphical interface may help users to define basic functional models).
- [REQHDSYST-010.2] : very basic consumption models defined by the user can be used without needing to select an existing consumption model from a library.
  The format of these basic consumption models must be like the one of the meta-model to be used by the estimators.
  (a graphical interface may help users to define them).

3) Simulation tools :

The user could dispose of simulation tools required to allow the verification of functional properties of a modeled system.
A set of open-source tools may be provided by default with the delivered versions of the software platform. (These are existing tools, not tools to be developed during this project.)
The aim is to provide a minimal base of tools to work with the Open-PEOPLE platform.

Requirements :

- [REQHDSYST-015] : The software platform will be delivered to users with a set of simulation tools given by default. Open-source, current and well-known tools will be provided.

7.1.3 Additional "Nice To Have" functionality

1) Templates :

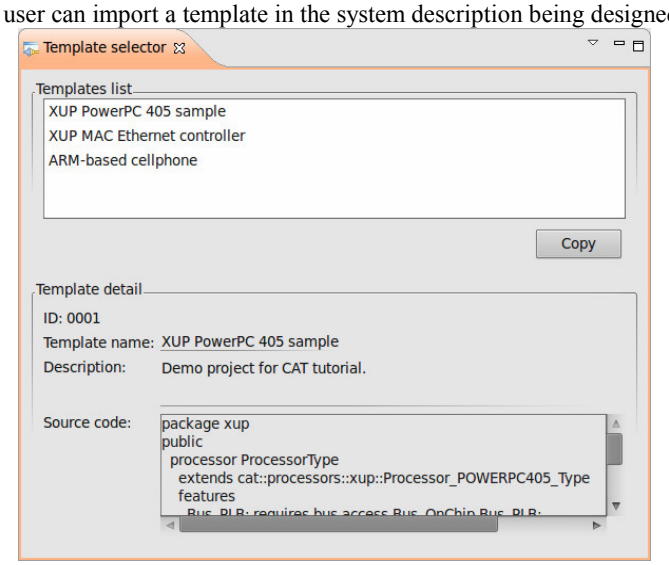
To ease the building of components systems description, templates could be provided for existing hardware components. (It could be a set of predefined template models for each development board of the hardware platform).
Templates could be stored in a library, put on the user computer for a local usage or accessible on a remote server.
The software platform would include the needed user interface for allowing the user to choose a template, to download it in his description. This (or another) interface could be equally used to add, or modify the proposed templates in the library.

Requirements :

- [REQNTSYST-000.1]: The software platform will allow the use of templates for system description building

The software platform will include a user interface to access and import templates. (→ refer).

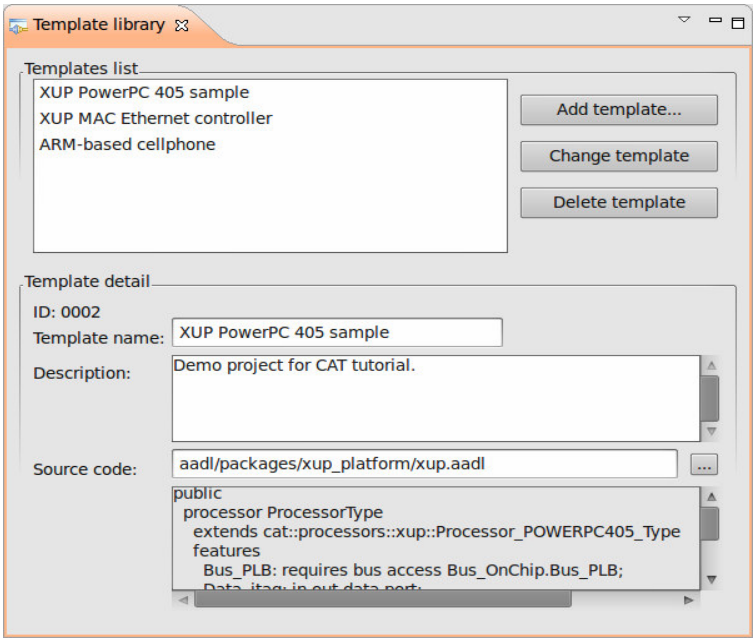
- A list of templates is proposed to the user;
- When the user selects one, the template details are displayed at the bottom of the screen;
- The user can import a template in the system description being designed.



— [REQNTSYST-000.2] : Templates will be stored in a library (local or remote location) (files or base),

The software platform will include a user interface to manage a library of templates. (-> refer)

- The user can see the templates already stored in the library;
- The user can add a template in the library;
- The user can modify an existing template;
- The user can delete one or several templates (after a deletion confirmation);



7.2 Power Consumption Analysis and Optimization functionalities

The H.264 application (video encoding) use case defined with all partners is given as example in the Annexe.

In the following, the functionalities are presented by a short description, also in term of requirements.
(they can be decomposed in one or several requirements).

7.2.1 Required functionalities

1) <u>Analysis and optimization :</u>

Power consumption analysis and optimization is provided by applying power and energy consumption models to the system architecture description model. It may be performed by existing estimation tools (mentioned in paragraph 6.3).
These tools should be included in the software platform by taking into account the features described earlier (such as modularity, ergonomics) and by being adapted to support off-line or Internet access mode of use.

It is expected to perform consumption analysis on platform composed at least of a processor, a memory and a bus.
(Indeed, buses have an important influence on the energy and power estimation of the underlying system).

Requirements :

- [REQPCAO-000.1] : The power consumption analysis will be performed by provided estimation tools of the software platform.
- [REQPCAO-000.2] : The estimation tools must be included in the software platform by taking into account features such as modularity, ergonomics.
- [REQPCAO-000.3] : The estimation tools must support off-line or Internet access mode of use.

2) Extensibility : integration of power and energy consumption models in the estimation tools :

The consumption models will have a well-defined format to be easily reusable by the different power and energy consumption analysis and optimization tools.
A consumption model is a set of input parameters and mathematical laws or a set of points used to calculate a consumption estimation.

It will be necessary to **define a meta-model of consumption models** in order to give a modeling of these laws (or models) under a specific format. (to be defined later).

The aim is that the consumption models can be used by the different estimators without having to modify the code of the estimators.

It would be useful to propose in the platform a user interface to define these laws and stored them under a predefined format (maybe under an XMI (XML Metadata Interchange) format that will ensure their persistence).

Requirement :

- [REQPCAO-005] : the power consumption model must be provided under a predefined format thanks to a meta-model.

3) Library of consumption models :

Consumption models will be stored in an extensible library.
A user interface will be provided in the software platform in order to access the library.
(These one could be hosted on the user computer for a local usage or accessible on a remote server).
We could for more secutiry verify the origin of the models stored in the library and consequently to set up a certification mechanism of the models with a signing mechanism (for example : "jar signing").

Indications could be given with each consumption models such as:

- the origin of the model
- a level of validity of the model
- a context on the consumption model development.

Requirements :

- [REQPCAO-010.1] : Consumption models will be stored in a library (local or remote location) (certainly a relational database)
  The software platform will include a user interface to access the library contents and allow users to easily integrate the consumption models in estimators.
  (-> a draft of user interface Figure 18: Consumption models library access).

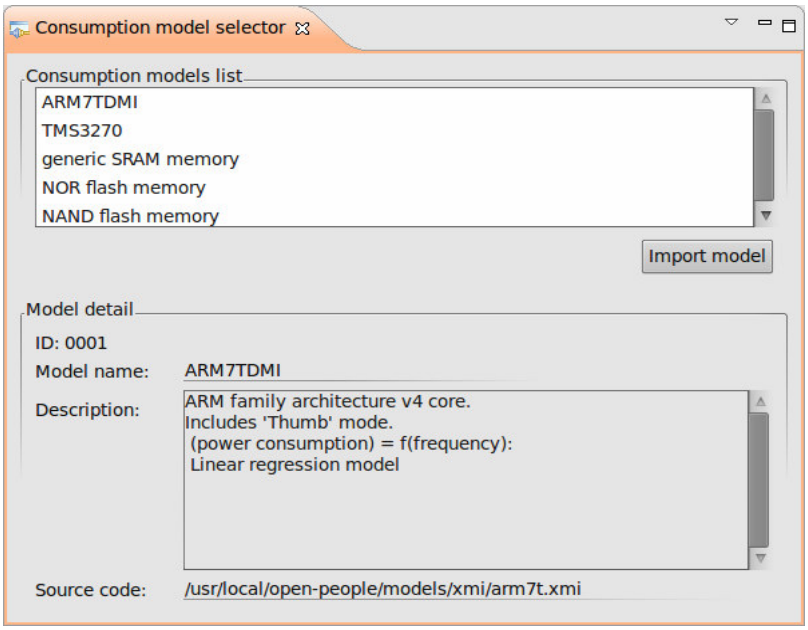


Figure 16 : Consumption models library access

- [REQPCAO-010.2] : The origin of the models stored in the library could be verified with a signing mechanism.

4) Model binding :

This is the binding of consumption models.

- [REQPCAO-015] : The software platform will allow users to choose the consumption models they want to integrate in the estimators to perform the power consumption analysis.

7.2.2 Highly desired functionalities

1) Granularity :

Requirement :

- [REQHDPCAO-000] : Users could define functionalities at system as well as component level, and equally for various refinement levels on component implementation details (source code, component circuit, component physical parameters, ...).

2) Dynamic reconfiguration :

In the software platform, power analysis and optimization functionalities would be also available for FPGAs.

For the power analysis, several possibilities of power estimation for FPGA can be considered :

- the use of the source code describing the FPGA application that can be analyzed by the platform.
- the use of functional blocks, to which VHDL or SystemC code could be associated (but not both together). (A refinement can be specified in SystemC, and then VHDL code or blocks of VHDL code can be simulated in SystemC, but no cosimulations SystemC and VHDL will be performed).
- the use of direct consumption measurements on the hardware platform (but this is only possible if the user provides a bitstream to the platform, obtained with his own FPGA development tools (the development tools are proprietary).

For these analysis, the dynamic reconfiguration should be extracted in order to define a consumption model of partial and dynamic FPGA reconfiguration.

For the power optimizations, operating system service(s) will be developed and optimization consumption model of dynamic reconfiguration will be proposed in the AADL operating system model.

Requirements :

- [REQHDPCAO-005.1] : For the power analysis of FPGAs, the software platform will allow to use three kinds of power estimation (source code, VHDL or SystemC code associated to functional blocks, direct consumption measurements on the hardware platform).
  It will imply to extracting the dynamic reconfiguration in order to define a consumption model of partial and dynamic FPGA reconfiguration.
- [REQHDPCAO-005.2] : For power optimizations, the software platform will allow to develop operating system services and will propose an optimization consumption model of dynamic reconfiguration in the AADL operating system model.

3) Uncertainty analysis :

The software platform should indicate to the user the degree of uncertainty for the power consumption analysis results. (The uncertainty depends on the refinement levels in the system description).
The uncertainty could be displayed in the view that will present results of power and energy consumption analysis.

Requirement :

- [REQHDPCAO-010] : The degree of uncertainty for the power consumption analysis results will be available for the user and can be displayed in the power consumption analysis results view.

(→ we can see an example of that in :).

4) Power presentation views :

- [REQHDPCAO-015] : The software platform should integrate views for components in a system description so that the user can check all important power and energy aspects of the components.

7.3 A brief description of the estimation tools to be integrated in the software platform

The software platform may integrate existing tools for power and energy consumption analysis and optimization.
The estimators currently listed for inclusion in the software platform are :

- CAT (Consumption Analysis Toolbox) with the two pieces of software it integrates (SoftExplorer, also called Power Estimation Tool and InterconnectExplorer), provided by the University of Bretagne Sud,
- and a software named GaspardLib, developed at the University of Lille.

7.3.1 CAT : Consumption Analysis Toolbox

Characteristics	
Language	Java, ATL
Compiler used	Sun, GCC
Supported OS	Windows, Linux
Code	750 classes
License	Not defined
Beginning of development	June 2008
Maturity	Alpha
Maintenance	Yes by Dominique Blouin
Architecture	Eclipse Plug-ins
Libraries used by CAT	Java libraries (jar), SoftExplorer et InterconnectExplorer executables
Software inputs	• AADL or DSL CAT (PADL) spécifications • Source code : C or ASM.
Software outputs	• AADL or DSL CAT (PADL) spécifications • consumption estimations.
Provided for integration	under format of Eclipse plug-ins or Java libraries (jar) in Open-source
Functionalities provided for the Open-People project	consumption estimation at system level including simple components

Description :

CAT uses as modeling tool the Open Source AADL Tool Environment (OSATE, provided in the CAT tool). The client application is the Eclipse Integrated Development Environment (IDE) plat-form.

CAT estimates the power consumption of hardware components and software components of a system (described thanks to the description language AADL), by using predefined power consumption models.
(A consumption model (a mathematical law or a set of points) is most of the time specific for a particular component).
In the scope of the Open PEOPLE project, functionalities will be developed to allow users to easily integrate the new consumption models into CAT (without adding code).

To compute power estimations for threads and processors, CAT uses SoftExplorer (that analyzes a C code to estimate the power consumption of a thread bound to a specific processor). To estimate the power consumption of ASIC bus devices, CAT integrates the InterconnectExplorer tool.
CAT can also perform estimation for composite components (components that have subcomponents), by calculating an average of the power consumption estimation.
It also offers the user the possibility to directly enter a power and energy consumption estimation for any component in the system.

CAT is provided for integration in the software platform in the form of Eclipse plug-ins and uses the Java language. (The source code of these plug-ins is provided. SoftExplorer and InterconnectExplorer are applications included in CAT).

The complete operation of CAT is given in the "cat user guide" document (by D. Blouin, UBS).
(This document is available on http://open.people.fr; Intranet).

7.3.2 GaspardLib

GaspardLib is an MPSoC architecture simulator.

Note : A point to be described in more details during the later steps of the project is the level of granularity implemented by each tools previously described for the power and energy consumption analysis. (Perhaps, one of the two could be used systematically for a coarse granularity of analysis and the other to refine the analysis.)

8 Power and Energy Consumption Models Development

8.1 Required functionalities

The following functionalities are all required in the software platform.
(The functionalities listed are described in term of requirements. They can be decomposed in one or several requirements).

1) Measurements :

- [REQPCMD-000.1] : The software platform will allow to measure different power consumption properties :
  - Static power of the Device Under Test (DUT) with all parts enabled or some parts disabled.
  - Dynamic power of the DUT with all parts enabled or some parts disabled.
  - For dynamic power, average and peak power must be measured.

- [REQPCMD-000.2] : Such measures, made on the hardware platform, could be saved and stored in a format that can be read and analyzed by the software platform during the power consumption model development step.

2) <u>Model development and validation :</u>

- [REQPCMD-005.1] : The software platform will allow to generate new power and energy consumption models from the measurements and will equally allow model validation. It could be made of an automatic comparison process between predictions from the model and measurements for a test application deployed on the hardware platform. Uncertainty analysis would be performed afterward.

- [REQPCMD-005.2] : These features should be provided by a dedicated Eclipse plug-in whose graphical user interface could look like this :

  - after opening a set of raw values (measured on the hardware platform), the set of values can be directly seen and edited on a first view (Figure 19: visualization and edition of raw data (first view)) :

Raw data

12 values

Var1: Frequency (MHz)	Var2: Power Consumption (mW)
100	50
150	69
200	88
250	109
300	128
350	153
400	171
450	192
500	208
600	264
700	312
800	368

Figure 17 : visualization and edition of raw data (first view)

  - the data can be graphically shown, as scattered points for raw values, plus geometric figures (like lines) for statistical model adjustment in a second view (Figure 20: graphical representation of raw data and statistical adjustment (second view)) :

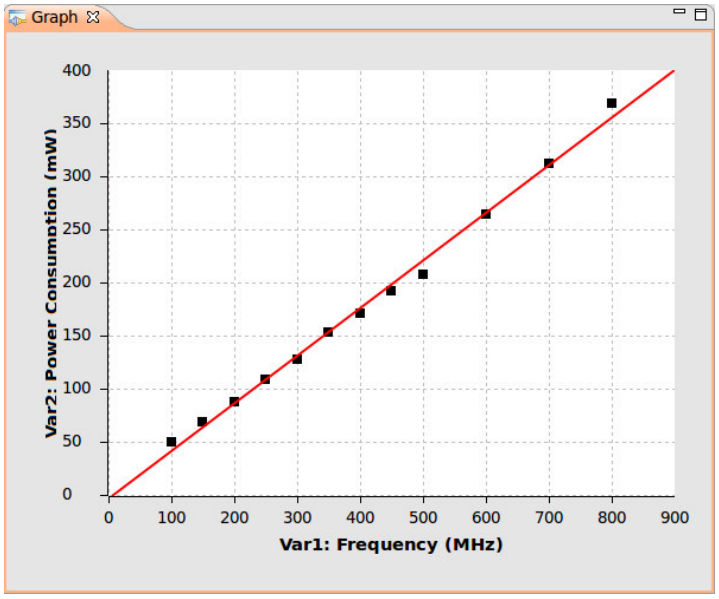


Figure 18 : graphical representation of raw data and statistical adjustment (second view)

- finally, models can be chosen (between "simple arrays" and different statistical adjustments), and their parameters refined and exported in a third view (Figure 21: calculation and optimization of statistical adjustment (third view)) :

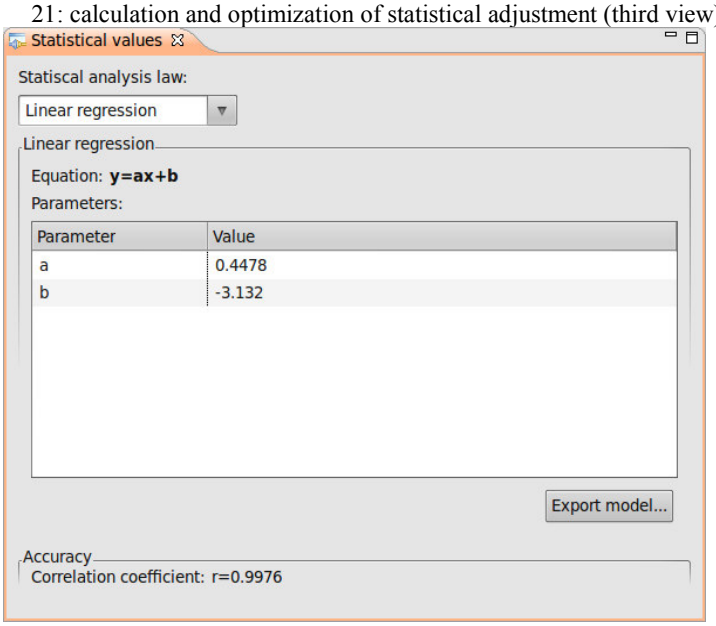


Figure 19 : calculation and optimization of statistical adjustment (third view)

(Note : Of course, modifications made on parameters will be propagated on the graph of the second view, so as to make graphical fitting of models in development easy; accuracy values shown in the third view should prove very useful, too).

3) Model exportation :

- [REQPCMD-010] : The software platform will allow to export the power and energy consumption models under a predefined format suitable for integration in power analysis and optimization tools. (→ Refer requirement [REQPCAO-005] defining a meta-model).

4) Export model in a persistent storage :

- [REQPCMD-015] : The developed consumption models must be stored in a (relational) database to be easily available for other software application tiers. The software platform will include a user interface to manage consumption models storage. (This consumption models database accessibility is described in requirement [REQPCAO-010.1]).

8.2 Use case diagrams

From previous delivrables, we can build the high-level use case diagram presented in Figure 22 hereafter. In this figure, PCMD features are represented by the “create QAML Model” use case, while PCAO features are represented by the “Perfom Design Space Exploration” use case. Each one of these use cases is divided into more “sub-level” use cases. Furthermore, D4.1 define the meta models to create and persist data: these meta models involve some of the functionalities described below.

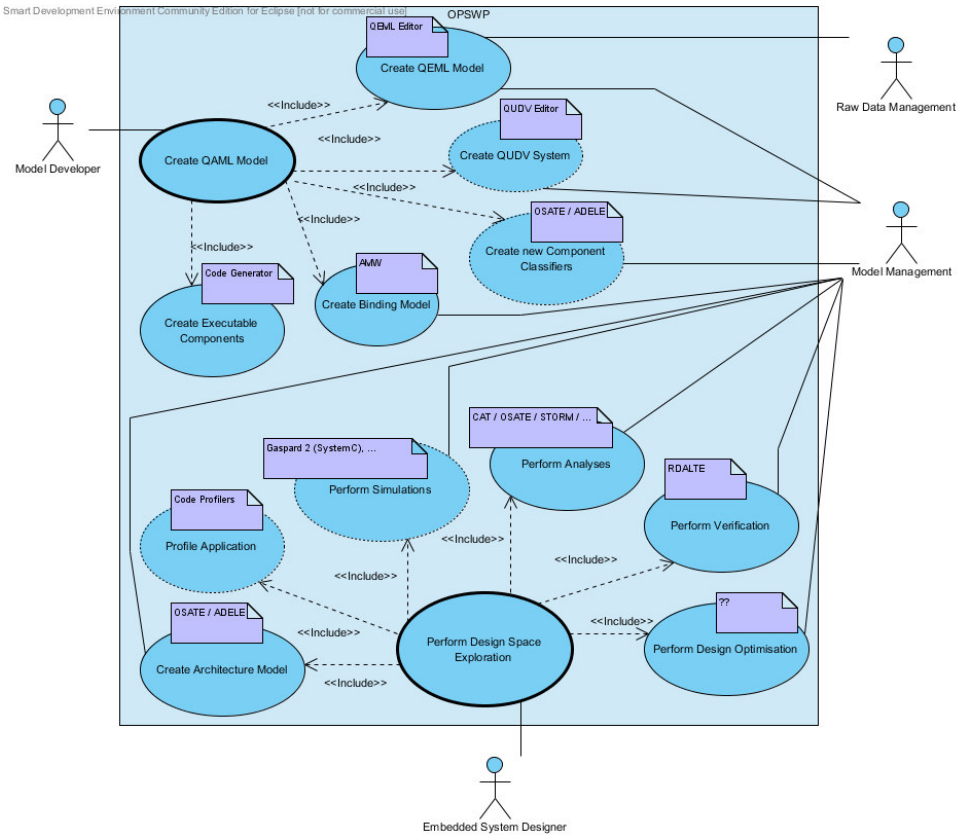


Figure 20 : A hight level use case diagram for thr Open-PEOPLE Software Platform (OPSWP)

8.2.1 Create QUDV system use case

The QUDV model is defined by the meta-model described in section 5.2.3 – Physical Quantities Concepts – of delivrable D4.1. The goal of this QUDV model is to define units and quantities; we can thus attach to it the following actions:

- Create an Unit
- Create a Quantity
- Delete an Unit
- Delete a Quantity

Because we don't want to force the user to explicitly create every model object, we try to automatize as much as possible objects creation. Firstly, one can notice that in this model, there is an almost complete symmetry between quantity-related notions and unit-related notions, although the unit-related part of the model is more complex than the quantity-related part. Consequently, quantities can reliably deduced from their related units; thus in our UI, the quantity creation task will be dependent on the corresponding unit creation task.
Multiples- and fractions-related prefixes (kilo, milli, etc.) are widely-known international standards, so we decided not to make making any user interface to manage such prefixes.
The LinearConversionUnit class is a particular case (i.e.: a subclass) of the AffineConversionUnit class, so we will – by default – only create AffineConversionUnit objects. Note also that

we won't support creation of GeneralConversionUnit objects, since there will be no use for such conversions in the context of the Open-PEOPLE project.
Thanks to the above study of the use cases for this meta model, we can now focus on the following, more restricted list of actions:

- Create an Unit:
  - a Simple Unit,
  - a Derived Unit (derived from a base, simple Unit),
  - an AffineConversionUnit (derived from other pre-existing Unit(s));
- Create a Quantity (this quantity being obligatorily related to an already defined Unit).

8.2.2 Create QEML model use case

The main use case for the QEML model is the creation of energy consumption models. Since this is a complex task, we subdivide this use case in two different sub-tasks, as shown on the diagram hereafter:

1. energy-consumption law creation
2. energy-consumption law composition

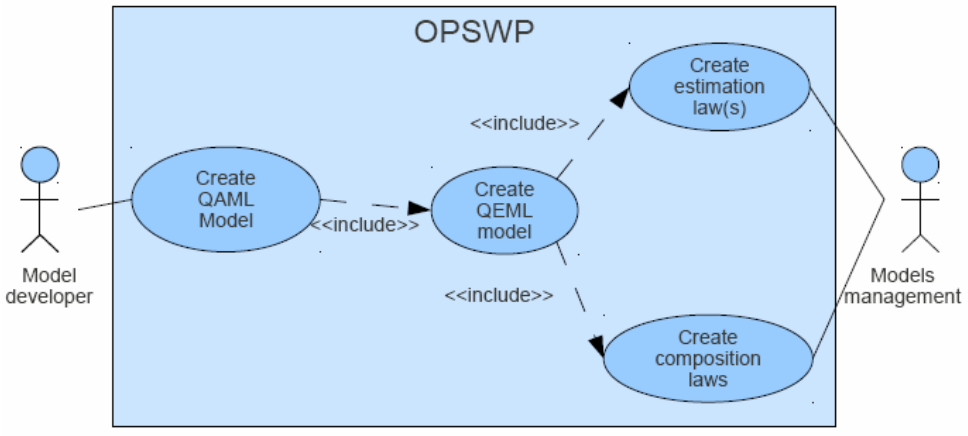


Figure 21: Refinement of "Create QEML model" use case

8.2.2.1 Create estimation law(s) use case

Summary : this use case allows a model developer to create one or more mathematical laws with several kind of quantities.
Actors : Model developer (principal), Models management (secondary)
Precondition : a QEML project is created

Main script

1. The Model developer requests the creation of a law;
2. The OPSWP asks the Model developer to describe the new law, with its domain of definition and its uncertainties, if applicable;
3. The Model developer gives the new law description;
4. The OPSWP asks the Model developer to define the quantities and units used by the new law's variables;
5. The Model developer chooses a quantity and the related unit for each variable;
6. The OPSWP checks if a quantity/unit couple is defined for each variable;
7. The OPSWP acknowledges that all variables have their quantity and unit defined.

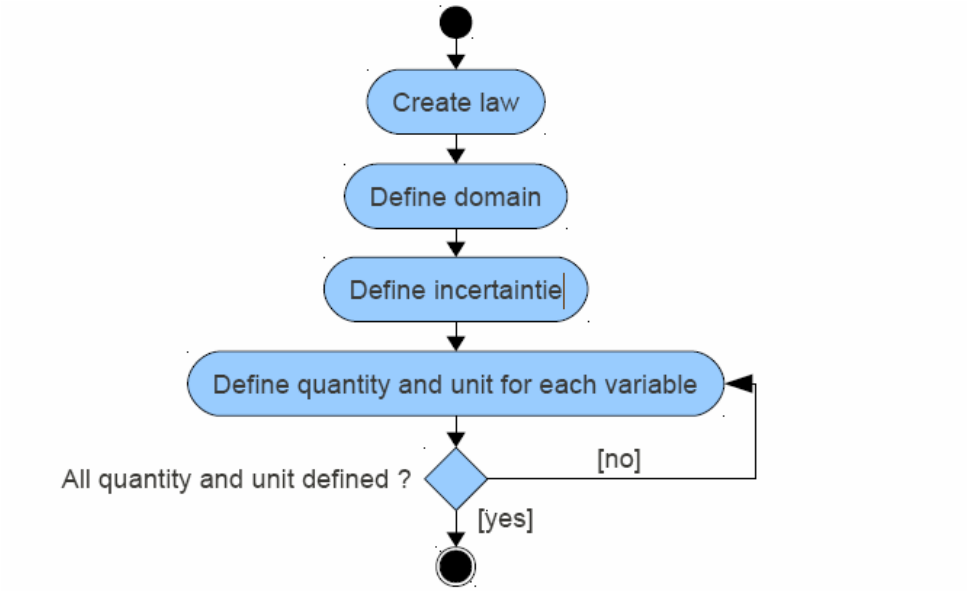


Figure 22 : Activity diagram of "create law" use case

Alternate script
A1: Some variables don't have their quantity and unit defined
The script resumes after step 6 of the main script:

7. The OPSWP notices than one or more variables don't have their quantity and unit defined, so it asks the Model developer to correct this problem;
8. The Model developer defines a quantity/unit couple for those variables which don't already have one;

The script then returns to step 6 of the main script.
Post-condition: the new law is created.

8.2.2.2 Compose laws use case

Summary : this use case allows a model developer to create one or more mathematical laws to aggregate quantities (of the same kind)
Actors : Model developer (principal), Models management (secondary)
Precondition: at least 1 preexisting law must be created before

Main script

1. The Model developer requests a new law creation and chooses the laws to be composes
2. The OPSWP asks the Model developer to give the new law
3. The Model developer gives the new law description, made by aggregating the chosen “base” laws;

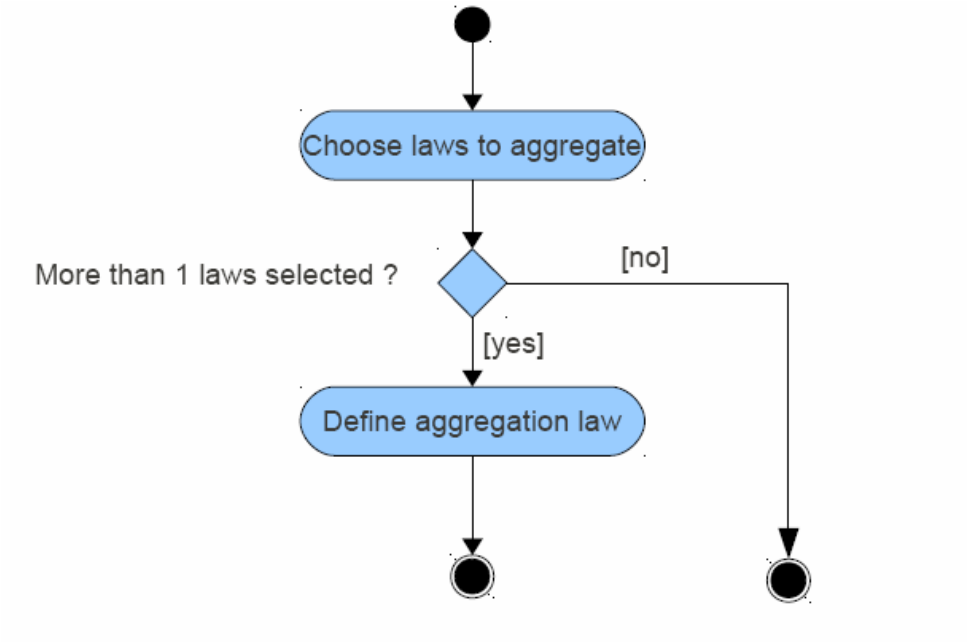


Figure 23 : Activity diagram of "compose law use case"

9 User interfaces

➔To allow the users to process energy and power consumption analysis and optimization, the software platform will include :

- graphical interfaces to allow the system designing,
- graphical interfaces to associate or consults properties on components.
- graphical interfaces to allow analysis and optimization.
- views to display the results of energy and power estimation.

The software platform will thus include the kind of interfaces that use the estimation tools CAT or GaspardLib.

Preview of these kind of user interfaces in CAT :

1. Components system can be designed in graphical way :

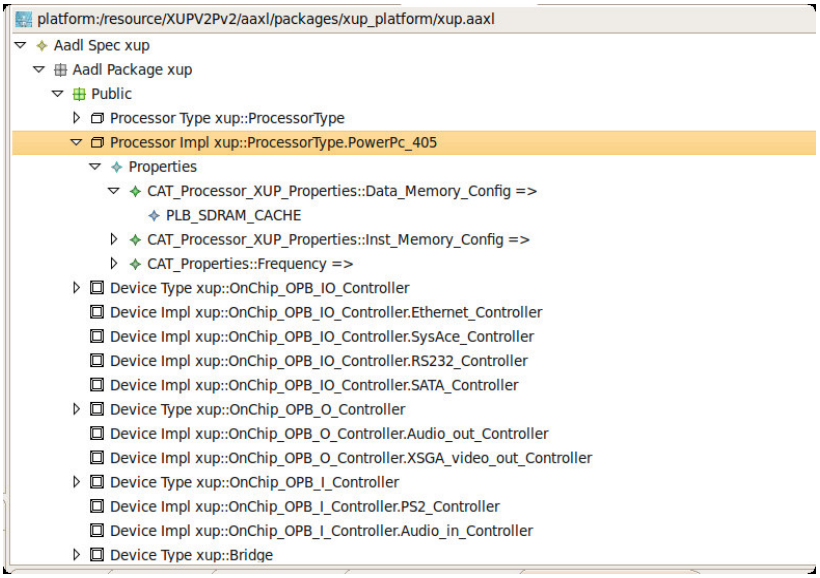


Figure 24 : CAT AADL object tree editor window

2. Components system can be designed in textual editor :

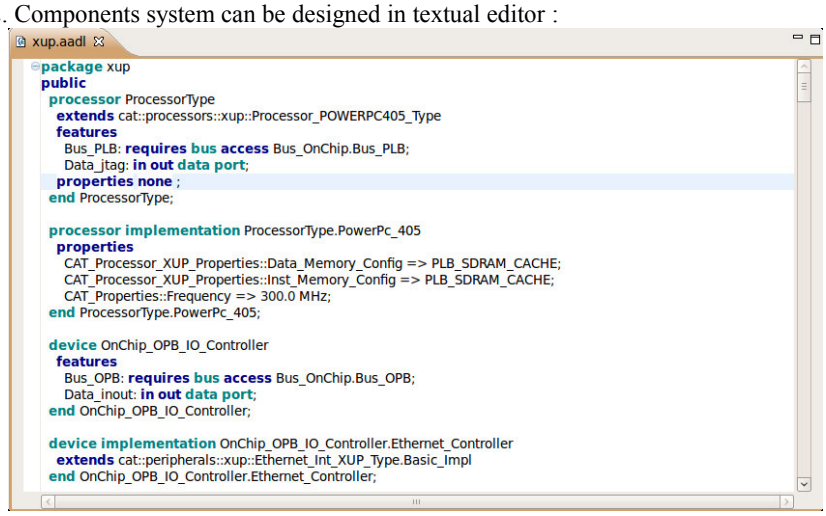


Figure 25 : CAT AADL textual editor window

3. Components properties :

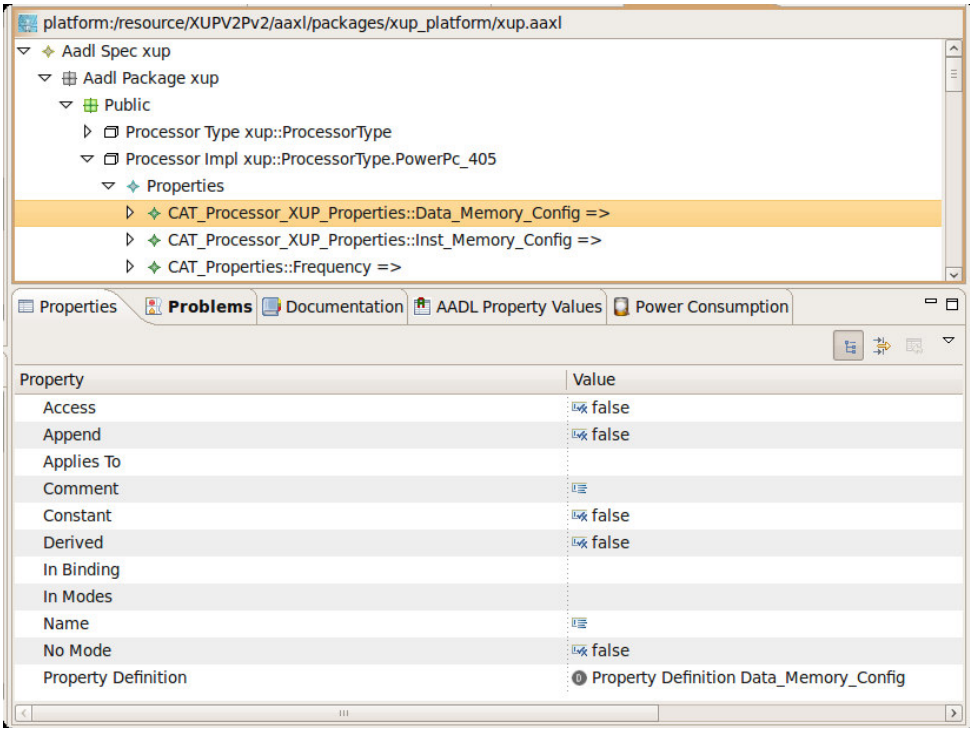


Figure 26 : Component Property View

4. AADL properties values :

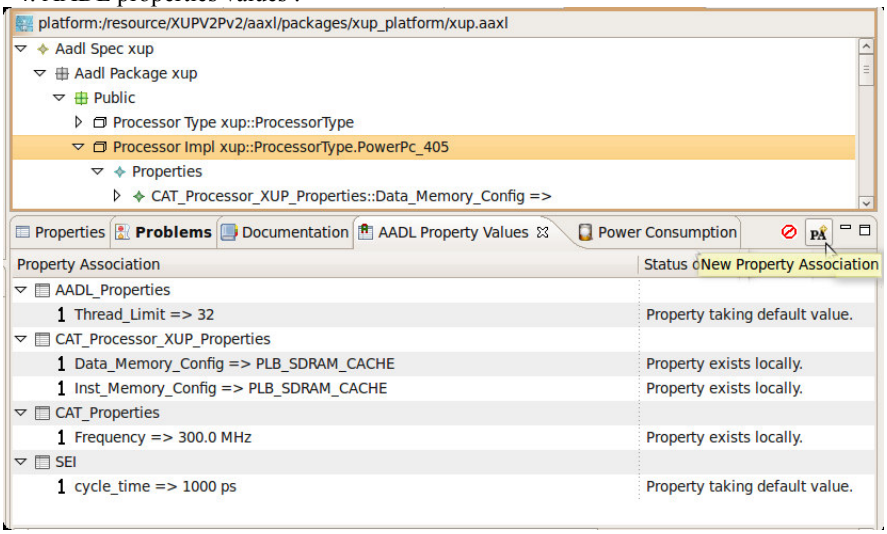


Figure 27 : AADL Property Values View

5. A power consumption results view :

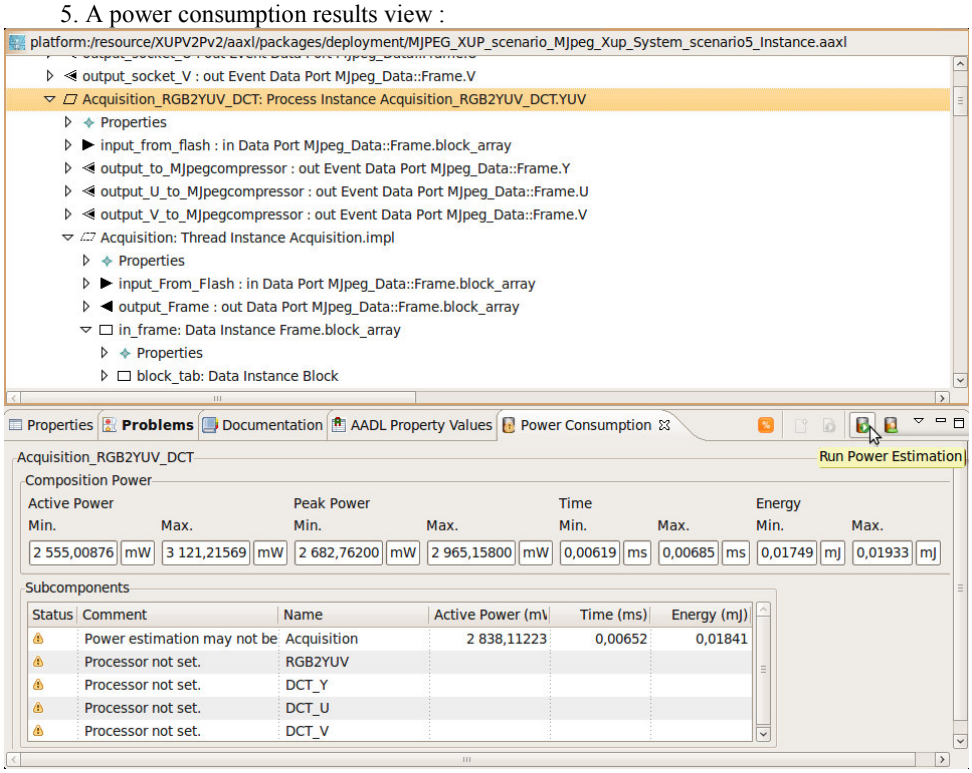


Figure 28 : Example of CAT Power consumption view

6. Power consumption results view with uncertainty

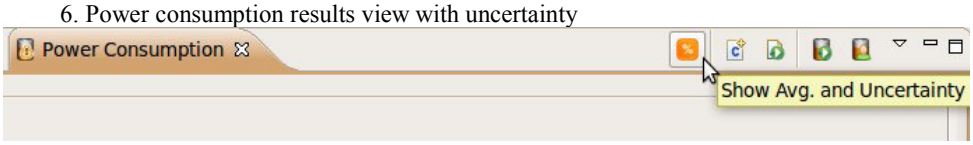


Figure 29 : Uncertainty option in results view

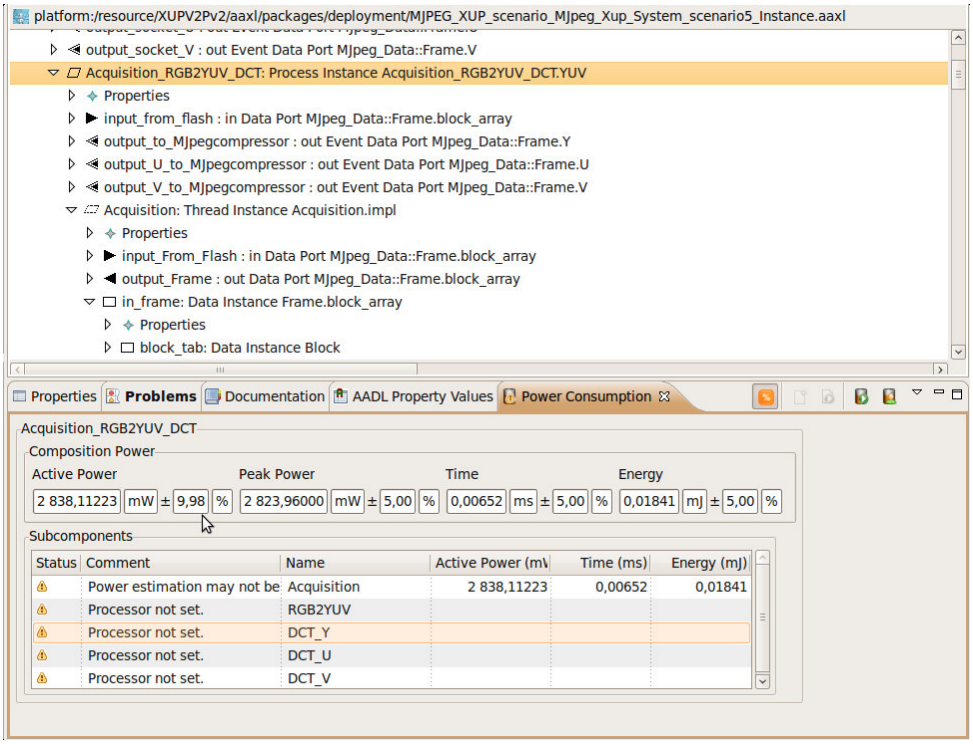


Figure 30 : Power consumption results view : display of uncertainty

➔ **Other interfaces** : (They have been already presented with the description of the functionality to which they correspond).

- The software application will provide the user with a **configuration** interface to set up the access mode and to change from off-line mode to Internet access mode : Refer [REQ-GNFCT-000].

- The software application could provide **preference pages** for the users : Refer [REQNTFCT-010].

- The software platform will include a user interface to **access and import templates** : Refer [REQNTSYST-000.1].

- The software platform will include a user interface to **manage a library of templates** : Refer [REQNTSYST-000.2].

10 Network protocol and bases

This paragraph gives some indications on which network protocol will be used by the software platform and also on databases that can be used when the software platform will be implemented.

- The software platform will use network protocol HTTPS protocol.
- This requires the use of authentication certificates : a Thawte certificate will be used for remote access to our server in INRIA Nancy (www.open-people.fr).

- The software platform should have access to relational databases (such as PostGreSQL or MySQL that are the databases that would be retained).

  We plan to implement a mechanism of synchronization between local bases and the base stored on our server.

  The local bases will store among others, the confidential data that can be : power consumption models, measurements obtain on hardware platforms.
  (The confidential data can become shared data according to the choice of their owner).

11 Tools provided with the software platform

List of proposed tools :

Tool included in plug-in available with the estimation tool CAT :

GCC (GNU Compiler Collection) modified version from GCC 4.2.0. :
- Modified and provided by the Lab-STICC (UBS) with the CAT estimation tool,
- necessary for the processors power consumption estimation,
- used to construct a model of Control Data Flow Graph (CDFG) that represents the functionalities on which consumption models are based.

(Model that is used by a tool of architecture synthesis (GAUT) and equally to perform the transformation from C code to AADL).

External tools provided with the software platform :

1. GCC (GNU Compiler Collection) :
- more precisely **G++** (C++ Compiler of GNU Compiler Collection) :
  For the high-level SystemC description of hardware components (such processors, memories, bus, ..) or IPs (Intellectual Properties).
- and **C Compiler** of GNU Compiler Collection :
  For C source code compiling.

2. IPs compiler :
- To compile the SystemC simulation models for IP coming from the external library **SocLib**.

(tool distributed as a free software by SocLib).

3. Cross-compiler tools :
- generated from standard GCC,
- to execute an application on a simulator that integrates an embedded processor,

- one tool needed from each type of processors (such as PowerPC, MicroBlaze, ARM).

4. Valgrind :
- A suite that includes a memory error detector, threads error detectors, a cache and branch-prediction profiler, a call-graph generating cache profiler, and a heap profiler,
- it runs on the following platforms: X86/Linux, AMD64/Linux, PPC32/Linux, PPC64/Linux, and X86/Darwin (Mac OS X),
- used for the validation,
- open-source under the GNU General Public License, version 2.

5. GHDL :
- A VHDL simulator, using the GCC technology, to compile VHDL files and creates binaries for simulations,
- **provided as a default VHDL simulation tool** with the software platform,
- currently, GHDL is only available for the GNU/Linux system and for Windows,
- under GNU General Public License.

6. GTKWave :
- A digital waveform viewer for examining the output of various simulators. It supports among others, standard Verilog VCD, or GHDL Waveform output format under which GHDL can generate a waveform file,
- **provided as a default tool** with the software platform,
- GTKWave is only available for Linux and Windows (as a Win32 application or via Gygwin),
- under the GNU General Public License.

12 Conclusion

This document :
- defined the architecture of the software platform : a standalone application that supports off-line or Internet access mode to use the data necessary to its work and that lets the user free to change and configure this access mode.
- defined the perimeter of the software platform by describing the required functionalities that are expected for first version of the software platform and also the additional functionalities that could be added in future versions.
- indicated and given a preview of which user interfaces the software platform shall provide.

13 Annexe : Use case : example of the H.264 application (video encoding).

1) Complex system descriptions build by the user must be decomposed into functional blocks.

→ These ones must be written with the description language AADL. For that, a graphical tool for AADL (Architecture Analysis and Design Language) modeling must be available in the software platform. (For example a tool like ADELE (a system architecture graphical editor based on AADL usable with or without TOPCASED), can be provided in the software platform).

→ or possibly, the application described in C code must be translated to AADL, that is to say to find the functional boxes from the C code.

2) Code is associated with each functional box : C code, VHDL code or possibly a material functionality (example of an IP).

3) Constraints are associated with the functional boxes (they are independent of the application description). (An example of constraints for an image would be : its size, a pixels array, number of images per seconds, real time, latency).
The modeling of the constraints needs the use of AADL analysis tools such OSATE.
➔(The tools are provided in the software platform).

4) It is necessary to define interfaces, described in requirements documentation and to take into account the use of the operating systems with or without DVFS (Dynamic Voltage and Frequency Scaling)).

5) The deployment model (in AADL) is enriched with the energy information inside the platform and with static analysis information by the use of others external tools.
(For example, the deployment of an application on the hardware platform consists in binding processes to memories, threads to processors and connections to buses, and for operating system in binding process and threads to the memory and processor).

The components performances will be compared with the applications requirements.
Results must be analyzed and parameters influencing power and energy consumption automatically extracted by the estimation tools used. (These values are afterward propagated in power consumption calculation).
→ The displayed results are for example : power, energy, current, consumption profile
➔**The platform provides views to display all the results obtained from the power consumption analysis.**

6) Case of SystemC (C++)/VHDL :
SystemC :
The functional description is always made in AADL. A tool will automatically translate the description into SystemC (the SystemC model of the components could be obtained with coarse or fine granularity).
➔In the platform, SystemC models are provided with a simulation tool and the platform realizes the interface with this simulator.
(a gcc compiler can be delivered with the software platform, with a prerequisite in the case of Windows environment (use of Cygwin)).

OS, OS services, peripherals must be taken into account in order that the simulation reflects the target plat-form.
→ First approach to take into account :
_ transformation from ADDL to SystemC (with refinement of some parameters)
_ results are send back to the platform to be stored in the AADL model.

7) Case of VHDL (for FPGA) :
As previously stated, the system is described with functional boxes (including processes) in the AADL language. The VHDL code is then manually associated to the functional boxes. Then the system and its deployment are modeled with the AADL language and analysis and estimation methods are applied.
(→ refinement can be done in SystemC and then code in VHDL on the FPGA
→ or VHDL blocks can be simulated in SystemC
but SystemC and VHDL co-simulations will not be allowed).

➔In the platform, AADL/FPGA models, deployment models of a thread on the FPGA and consumption models are provided also a simulation tool (for SystemC). The platform provides the interface with this simulator .
(For the FPGA, simulation tools are proprietary and can't be distributed with the plat-form).