The Kitalpha project is a proposed open source project under the PolarSys Top level Project.

This proposal is in the Project Proposal Phase (as defined in the Eclipse Development Process) and is written to declare its intent and scope. We solicit additional participation and input from the Eclipse community. Please send all feedback to the Eclipse Proposals Forum.

Background

In system and software engineering, a common need during the phases of analysis, design, and implementation, is to describe a system. Several standards have been established to define a shared notation or concepts for system description. For instance:

- The objective of UML is to provide system architects, software engineers, and software developers with tools for analysis, design, and implementation of software-based systems as well as for modeling business and similar processes [UML 2.5, FTF].
- The ISO/IEC/IEEE 42010 standard defines requirements on the description of system, software and enterprise architectures. It aims to standardize the practice of architecture description by defining standard terms, presenting a conceptual foundation for expressing, communicating and reviewing architectures and specifying requirements that apply to architecture descriptions, architecture frameworks and architecture description languages.

Multiple tools implement such standards. At Eclipse, UML2 and Papyrus implement the UML standard. While it would be possible to implement the ISO/IEC/IEEE 42010 standard with a UML tool, no Eclipse or PolarSys component has natively implemented it yet.

Scope

Kitalpha is a modeling component which implements the ISO/IEC/IEEE 42010 standard for system description in system and software engineering. It provides both a development and runtime environment to create and execute rich MBE (model-based engineering) workbenches (e.g., edition with diagrams, documentation, import/export, model transformation / analysis / validation) for system / software architects and engineers in small- to large-scale projects.

Conforming with this standard, an MBE workbench is an architecture framework which aggregates viewpoints clearly separating concerns (e.g., performance, safety, security, cost). A viewpoint is an engineering extension which comes with its own metamodels, representations (e.g., diagrams, tables, user interfaces), rules (e.g., validation, analysis, transformation), services and tools to address an engineering specialty. Consequently, an MBE workbench is the result of a flexible assembly of core viewpoints extended by new ones which are, in the context of co-engineering, appropriate and valuable for specialty engineers. The set of all the viewpoints defines the complete description of a system.

To build MBE workbenches, a lesson learned at Thales is that designers must be autonomous in creating and maintaining their own viewpoints, without coding. Developers can enrich them afterward, for instance for algorithm implementation. To meet this requirement, Kitalpha offers a development environment made of DSLs (Domain-Specific Languages) to assist designers and

developers in their architecture frameworks and viewpoints development activity activities. For instance, textual editors make it possible to declare viewpoint metamodels, user interfaces, diagrams, or services. From those DSLs, generators build all the architecture framework and viewpoint artifacts. For example, the declaration of diagrams using DSLs becomes the technical description of Sirius diagrams. During the stages of edition with DSLs and generation, the notion of target application is introduced to manage the variability of environments in which the artifacts are to be deployed and executed (e.g., DSL vs. UML, CDO vs. XMI environments).

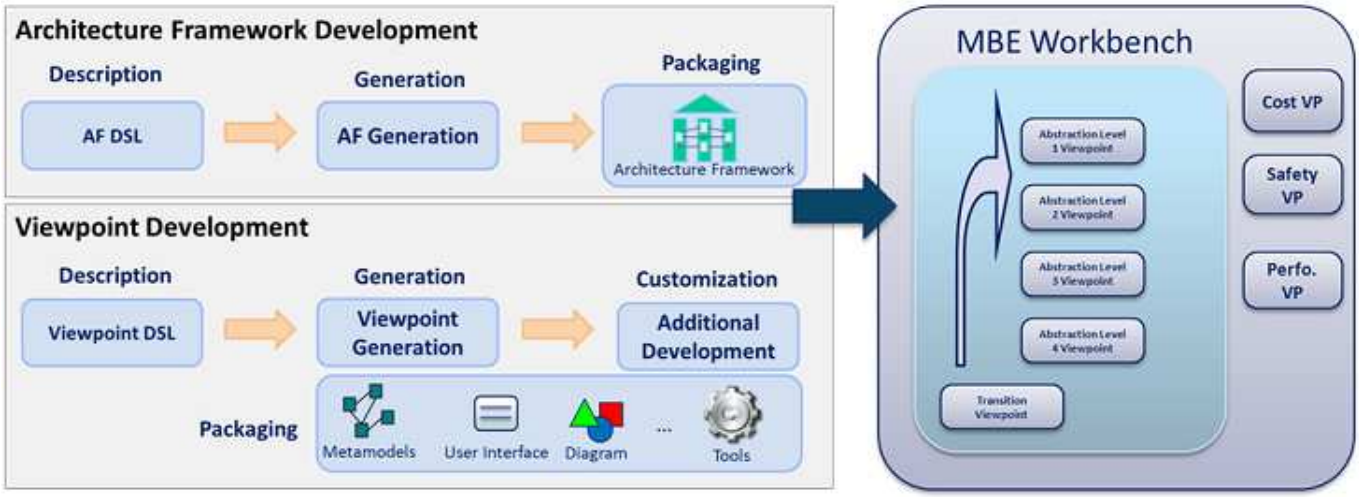

Figure 1. Development Process Overview

The Kitalpha development environment also provides basic components to develop complete MBE workbenches, for instance for the functions of import/export (e.g., for data exchange with MBE workbenches or specialty tools), model transformations with traceability, or html documentation generation.

Kitalpha potentially has the ability to implement architecture framework standards (e.g., TOGAF/MODAF) but also proprietary method or domain workbenches.

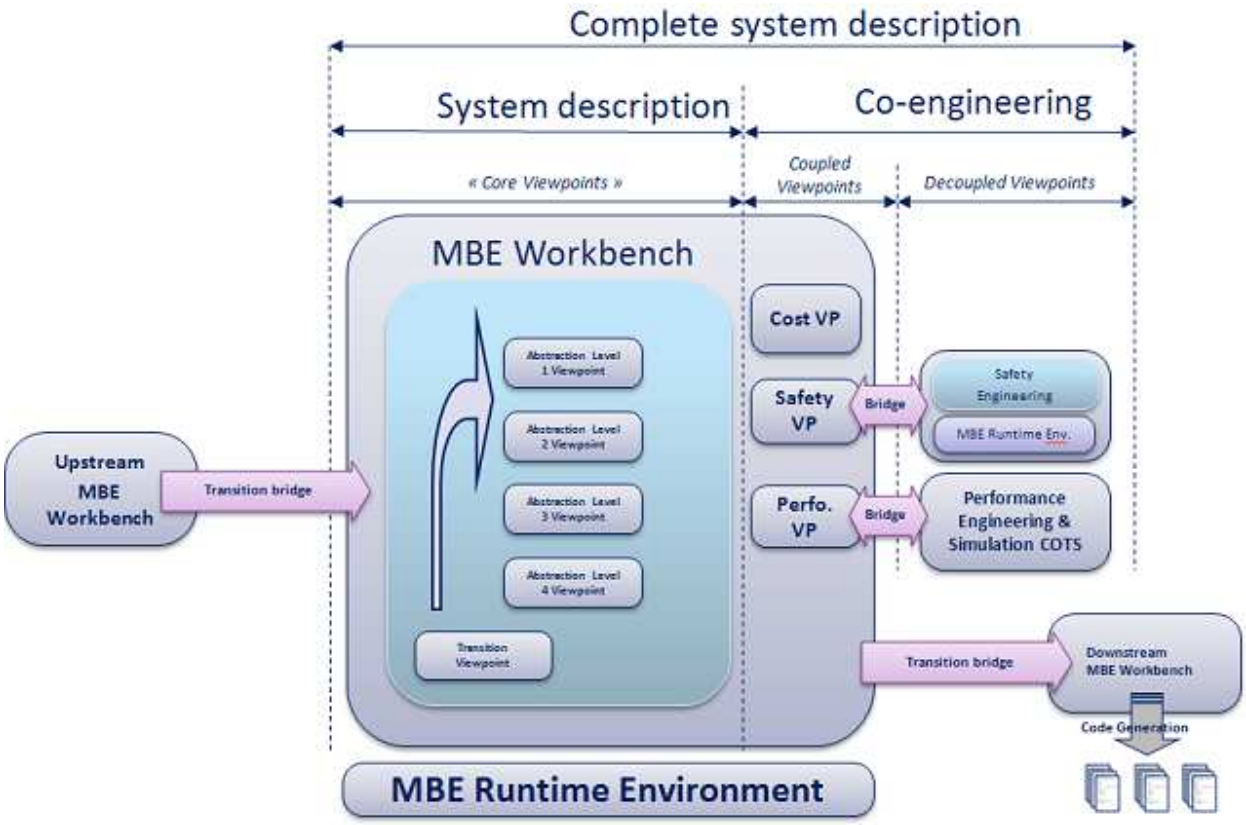

Figure 2. Illustration of MBE workbench

Description

This section describes the key features of Kitalpha

Anatomy of Kitalpha

Architectural position of Kitalpha

Kitalpha can be considered as a meta-tool. The following picture presents the different working contexts with regard to Kitalpha.

1. At the top, stakeholders describe a system architecture,
2. An MBE workbench enables designers to describe system architecture,
3. Kitalpha is a workbench allows developing and executing MBE workbenches,
4. Finally, Kitalpha is built atop Eclipse and mainly uses modeling components.

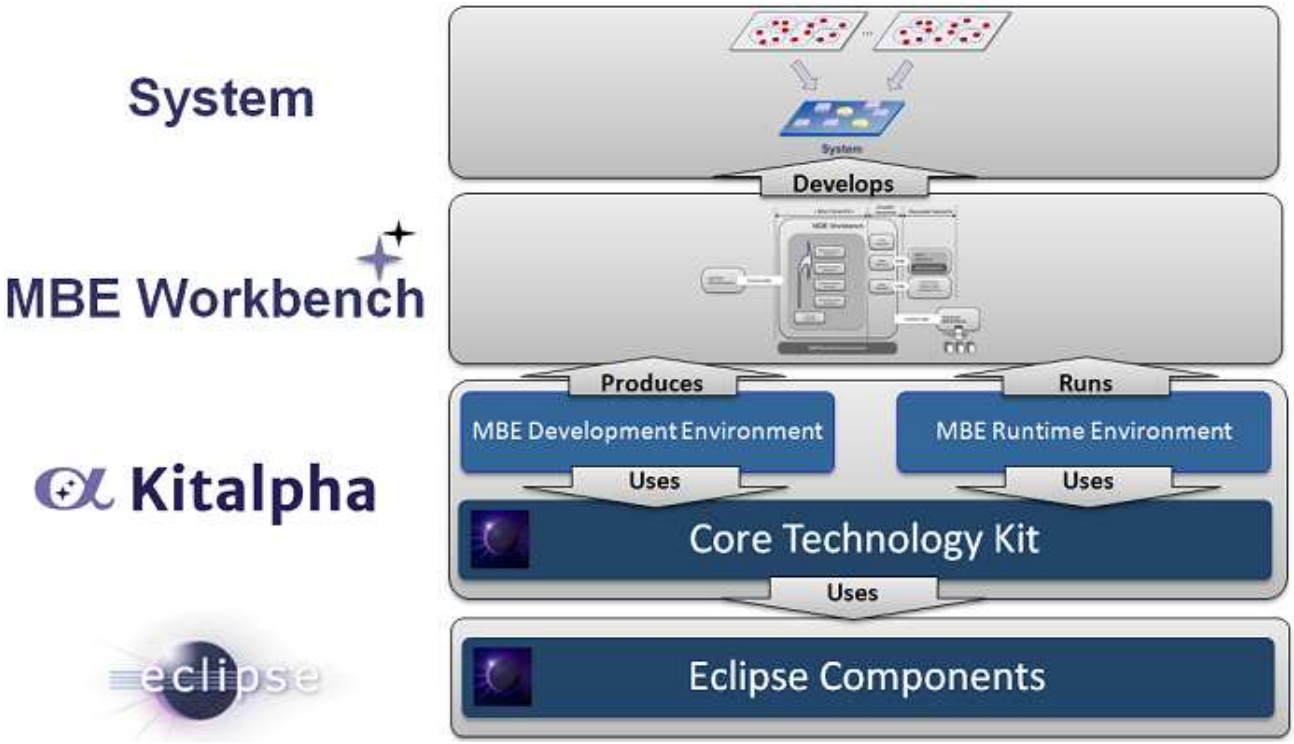

Figure 3. Position of Kitalpha

Elements of internal architecture

Kitalpha is divided into two layers:

- An *Engineering Layer* which is a set of components and services for developing and executing MBE workbenches. This layer contains all the architecture framework / viewpoint metamodels, DSLs with their representations, and services (Cf. section about the Kitalpha services).
- A *Core Technology Kit (CTK)* which is a set of technical components and frameworks required by the Engineering Layer.

The CTK is dedicated to host a set of added-value technological components which are not provided by Eclipse components or to adapt existing tools for a specific purpose (e.g., adaptation of the standard EGF factories). A substantial tool or set of tools can be grouped in a Kitalpha sub-project for a better partitioning and decoupling of the sub-components development lifecycles.

Precision of the ISO/IEC-42010 standard

The ISO/IEC-42010 standard is intentionally general in order to be open and declined into multiple implementations. The following parts present the choices made to implement the standard.

Kitalpha viewpoint

In compliance with the standard, a Kitalpha architecture framework aggregates viewpoints. A Kitalpha viewpoint is defined as:

- A set of metamodels
- A set of notations (e.g., icons)
- A set of representations (e.g., textual, graphical)
- A set of rules (e.g., check, transformation)
- A set of services
- A set of tools
- Moreover, a viewpoint inherits and aggregates viewpoints.

Kitalpha services

Kitalpha provides both development and runtime services to define, use and manage architecture frameworks and viewpoints.

Main services at development time are:

- For Architecture Framework (AF): 1) definition of an AF, 2) generation of AF artifacts, 3) packaging of AF artifacts with the viewpoints it aggregates.
- For Viewpoint: 1) definition of a viewpoint, 2) generation of viewpoint artifacts, 3) packaging of viewpoint artifacts.

Main services at runtime are:

- Core services:
    - System architecture description with an architecture framework and its viewpoints
    - Viewpoint Manager in order to monitor viewpoints
    - Activation / Deactivation of a viewpoint
    - Detachment / Attachment of viewpoint data
    - Migration of a viewpoint
- Additional services:
    - Versioning
    - Team Working
    - Reporting
    - Architecture Assessment
    - Test
    - Simulation

Why PolarSys?

By releasing and developing Kitalpha publicly, the objectives are the following:

1. Strengthening the PolarSys and Engineering Eco-system by enabling the creation of rich Model-Based Engineering workbenches at a very low cost.
2. Providing an open-source implementation of the ISO/IEC-42010 standard.
3. Being an enabler to provide viewpoints among the PolarSys community.

4. Unleashing the potential collaborations with other partners and projects in a healthy environment.

Relationship with other Eclipse and PolarSys Projects

The Kitalpha project is based on Eclipse Modeling projects, notably:

- Xtext to textually define architecture frameworks and viewpoints.
- Sirius is used at development time (e.g., for a graphical rendering of the architecture frameworks and viewpoints, or documentation) and runtime (e.g., for edition with diagrams in MBE workbenches).
- Ecore Tools 2+ for edition of viewpoint metamodels.
- EGF to mass-produce architecture framework and viewpoint artifacts from textual descriptions.
- Each aspect of a viewpoint description is generally implemented by at least one Eclipse / PolarSys component. For example, OCL can be used to express constraint rules.

Initial Contribution

The initial contribution includes:

- The development and runtime environment to create and execute MBE workbenches.
- The runtime environment limited to the core services listed in the section of Kitalpha services.

Legal Issues

Committers

All contributions will be distributed under the Eclipse Public License.

Project lead:

- Benoit Langlois, Thales Global Services

The following individuals are proposed as initial committers to the project:

- Benoit Langlois, Thales Global Services
- Boubekeur Zendagui, Obeo
- Guillaume Gebhart, Obeo
- Jean Barata, Thales Global Services
- Matthieu Helleboid, Thales Global Services
- Philippe Dul, Thales Services SAS
- Thomas Guiu, Soyatec

We welcome additional committers and contributions.

Initial Contributors

Thales will provide the initial contribution which has been developed in the context of the Sys2Soft (BGLE) and Crystal (Artemis) project and that is already deployed in an industrial context. Additional contributor of the initial version:

- Amine Lajmi, Itemis

Mentors

The following Architecture Council members will mentor this project:

- Cedric Brun
- Kenn Hussey

Interested Parties

The following individuals, organizations, companies and projects have expressed interest in this project:

- Altran
- Obeo
- Samares
- Soyatec
- Thales Global Services

Project Scheduling

- February 2014: code submission and IP review
- March 2014: 0.2 release
- June 2014: 0.3 release based on Luna

Changes to this Document

Date	Change
10-December-2013	Document created