Apollo

Overview specification

Confidential

Revision number: 1

Date:

Name: P. van der Velde

# Preface

Bla blab la

Contents

[Preface 2](#_Toc208673628)

[Disclaimer 2](#_Toc208673629)

[Introduction 2](#_Toc208673630)

[Architecture 3](#_Toc208673631)

[Interaction 3](#_Toc208673632)

[Core 3](#_Toc208673633)

[User interface 3](#_Toc208673634)

[Plug-ins 3](#_Toc208673635)

# Disclaimer

# Introduction

This specification will define the global architecture of Apollo.

Apollo consists of 3 parts

* core
* User interface (UI)
* At least one but probably more plug-ins

Apollo gets all its power through the plug-ins. The core or the UI are not capable of any data generation or processing. The plug-ins provide the data generation, storage and visualization capabilities. The core provides tools for the administration of the plug-ins and the UI provides the interface with the user.

Describe:

* Interaction between the different parts
* Core
* User interface
* Plug-in behavior

# Architecture

This describes the general architecture of the different parts

## Interaction

* The core provides general services that can be used by both the UI and the plug-ins. The UI will have access to more different services than the plug-ins will.
* The core provides two different external API’s that can be used at run-time
  + User interface API
  + Run-time plug-in API
* Furthermore the core also provides three non-run-time API’s
  + Generic plug-in API. This allows components to have their own plug-ins. Defines a series of standard rules that allow the core to locate and classify plug-ins.
  + Project plug-in API. Allows developers to write project plug-ins. Project plug-ins provide extra types of generators and visualizers(?). This API is based on the generic plug-in API.
  + Component plug-in API. Allows developers to write components. The plug-in API ensures that plug-ins can be found by the system.

## Core

The core contains the kernel (services, global command system, global data), the project system and the core UI system. All three parts provide API’s which are at least within the core generally available. To the outside only a part of the project system API and the full core UI API are available at run-time. On top of that both API’s are only available to specific external components.

The project API is only available to two different types of plug-ins. The project plug-ins use one specific part of the API and the feature & component plug-ins use another part. Each type of plug-in will only be able to use their specific part of the API.

The user interface API is only available to user interfaces and not to plug-ins.

## User interface

## Plug-ins

* What is expected from the plug-ins (see the plug-in specification doc for more specific details)?
* What can plug-ins do?
* What can plug-ins not do?
* What are the rules for plug-ins?