

Coalevo Architecture - Foundations

Table of contents

| | |
|--------------------------------------|---|
| 1 Synopsis..... | 2 |
| 2 Motivation and Requirements..... | 2 |
| 3 Design Foundations..... | 2 |
| 3.1 Java Platform Contributions..... | 3 |
| 3.2 OSGi Platform Contributions..... | 3 |
| 4 Related Documentation..... | 3 |
| 4.1 Architecture Overview..... | 3 |
| 4.2 Bundle Anatomy..... | 4 |

1. Synopsis

The Coalevo project is trying to create a new software platform for building and sustaining virtual network communities. This type of platform presents a relatively complex system and requires foundations that provide reliability, extensibility and scalability.

2. Motivation and Requirements

The main motivation of the Coalevo project is to create a software architecture that:

1. allows to tackle the problem of complexity using a *divide and conquer* approach (i.e. decomposition);
2. has a *design for change*, that allows to extend and change the system throughout its lifecycle; and
3. is open for integration with other systems.

Therefore, the main focus of the architecture is to fulfill the following requirements:

- modularity;
- extensibility; and
- interoperability and integrability.

3. Design Foundations

A modular software system that can be assembled or extended by components (i.e. software modules), requires:

- a language that permits the definition of interfaces for components;
- a mechanism to find and obtain instances of components with certain properties;
- a binary format that allows the component to be loaded at run-time;
- a mechanism that allows to create instances of components at run-time;
- a mechanism that manages different versions of components;
- conventions to use methods of component interfaces;
- a mechanism to navigate polymorphic types;
- a mechanism to reclaim memory that is no longer used; and
- a mechanism to communicate between components.

This list has been obtained from a study of existing component based systems in the context of an investigation on complex control systems that are confronted with increasing portion of software (around 80%) [R. Sanz, Pfister C., Schaufelberger W., and A. De Antonio, "Software for complex controllers", ch. 7, pp. 143–164, Springer Verlag, 2000].

Given the motivation and to fulfill the presented requirements, the architecture uses two main elements as base for the architectural design:

1. the Java platform; and
2. the OSGi framework.

Implementations of these two elements also form part of the stack that provides the run-time foundation for the Coalevo platform, as shown in the following figure:

At the architectural level, the individual contributions of each element provide the foundation for a modular software system.

3.1. Java Platform Contributions

1. a language that permits the definition of interfaces for components;
2. a binary format that allows the component to be loaded at run-time;
3. a mechanism that allows to create instances of components at run-time;
4. conventions to use methods of component interfaces;
5. a mechanism to communicate between components;
6. a mechanism to navigate polymorphic types; and
7. a mechanism to reclaim memory that is no longer used (i.e. garbage collection).

Note:

More information on the Java Platform can be found [here](#).

3.2. OSGi Platform Contributions

1. a mechanism that manages different versions of components;
 2. a mechanism that manages the lifecycle of components; and
 3. a mechanism to find and obtain instances of components with certain properties; and
- Additionally, OSGi also provides a well defined unit of deployment, called *Bundle* which is great for the management of complexity in the development process. As well as components that already provide basic services (for configuration, logging etc.) as part of the OSGi service compendium.

Note:

More information on the OSGi platform can be found [here](#).

4. Related Documentation

4.1. Architecture Overview

Please visit the [Coalevo Platform Architecture Overview](#) for more information.

4.2. Bundle Anatomy

Please visit the [Coalevo Bundle Anatomy documentation](#) for more information.