实验三 软件需求的跟踪分析

实验思路：

确定开源IDE，eslipse

选定需求：

Eclipse Dirigible是一个开源项目，它为正在运行的应用程序提供集成开发环境即服务（IDEaaS）以及运行时容器集成。

全部讨论文本：

**Overview**

Eclipse Dirigible is an open source project that provides Integrated Development Environment as a Service (IDEaaS) as well as runtime containers integration for the running applications. The environment itself runs directly in browser and therefore does not require additional downloads and installations. It packs all the needed components, which makes it self-contained and well integrated software bundle that can be deployed on any Java based Web Server.

Target applications built with Dirigible are atomic yet self-contained cloud based modules that cover end-to-end vertical scenarios. These applications comply with the Dynamic Applications concepts and structure.

**Dynamic Applications**

The overall process of building Dynamic Applications lies on well-known and proven principles:

**In-System Development** - known from microcontrollers in business software systems. The major benefit is working on a live system where all the changes made by a developer take effect immediately, hence the impact and side-effects can be realized in a very early stage of the development process.

**Content Centric** - known from networking, in the context of Dynamic Applications it proclaims that all the artifacts are text-based models and executable scripts stored in a generic repository (along with the related binaries, such as images). This makes the life-cycle management of the application and the transport between IT landscapes (Dev/Test/Prod) simple and straightforward. In addition, a desired effect is the ability to setup the whole system, only by pulling the content from a remote source code repository, such as Git.

**Scripting Languages** - programming languages written for a special run-time environment that can interpret (rather than compile) the execution of tasks. Dynamic languages existing nowadays as well as the existing smooth integration in the web servers make possible the rise of the in-system development in the cloud.

**Shortest turn-around time** - instant access and instant value became the most important requirement for the developers, that's why it is the major goal of our tooling.

In general, a Dynamic Application consists of components, which can be separated in the following categories:

Data Structures - the artifacts representing the domain model of the application. There is no intermediate adaptation layer in this case; hence all the entities represent directly the database artifacts - tables and views.

Entity Services - domain model entities exposed as web services, following the modern Web 2.0 patterns and scripting language capabilities.

Scripting Services - general purpose services for utilities, custom modules and extensions.

User Interface – pattern-based, generated user interfaces, as well as custom forms and reports based on the most popular AJAX frameworks or any other used by the developer.

Integration Services - in the real world there are always some external services that have to be integrated into your application - for data transfer, triggering external processes, lookup in external sources, etc. For this purpose Dirigible provides capabilities to create simple and dynamic routing services following the Enterprise Integration Patterns guidance

Documentation - an integral part of any application is its documentation. Dirigible supports authoring documentation content in the widespread wiki (confluence) format.

**Architecture**

Dirigible architecture follows the well proven principles of simplicity and scalability in the classical service-oriented architecture.

The design-time and the runtime components are well separated. They interact with each other through a dedicated repository where the only linking point is the content itself. At design-time programmers and designers use the Web-based integrated development environment based on the Eclipse Remote Application Platform (RAP). Leveraging this robust and powerful framework the tooling can be easily enhanced using well-known APIs and concepts - SWT, JFaces, OSGi, extension points, etc.

The Repository is the container of the project artifacts. It is a generic file-system like content repository built on a relation database (via JDBC).

On top are the Dirigible's containers for services execution depending on the scripting language and purpose - Rhino, jRuby, Groovy, Camel, CXF, etc.

The runtime can scale independently by the design time part, and even can be deployed without design time at all (for productive landscapes).

Depending on the target cloud platform, Dirigible can be integrated to use the services provided by the underlying platform.

需求变更：

2.6版本：

There are a few topics in this release:

* Enhancements on the "Templates" authoring capabilities
* Swagger integration for API discovery and test
* Enterprise JavaScript API new modules
* Generation services exposed as REST services
* Dependency references feature

2.7版本：

This release provides enhancements in the RAD area such as new templates Master/Details, Map, Launchpad, Cockpit, CRUD templates based on the expereimental declarative REST frameworks, new Themes, etc.

Another major step in this release is the Kubernetes setup with the necessary configurations for scalability as well as the implementation with Dirigible of a management tool for Kubernetes - ZEUS.

The third set of features is the "lean"/non-OSGi AppServer distribution - AIR, with all the related optimizations.

3.0版本：

This release will be a major technology change as well as simplification regarding life-cycle management of the projects.

* The RAP based WebIDE will be separated form the AppServer
* AppServer will go non-OSGi
* Light WebIDE will be promoted as default - Angular 1.x/Bootstrap with Orion Editor
* Production readiness
  + Deployment model - Kubernetes + Docker
  + Structured Storage (RDBMS) - PostgreSQL
  + Unstructured Storage (noSQL) - Cassandra
  + Messaging Service - ActiveMQ
  + Process Engine - Flowable
  + Indexing Engine - Lucene
  + JavaScript Engine - Nashorn with ES6 support and V8 as an option
  + CXF for RESTful services
* Project structure simplification
  + Plain Java modules for services
  + Webjars like modules for APIs and IDE plgins
  + Webjars like modules also for Dirigible's Applications

3.1版本：

* Custom Datasources support
* Database Repository support
* CMIS support
* Bookstore template
* Job Template
* Listener Template
* Documents Perspective
* Minor bugfixes and improvements

3.2版本：

* Flowable engine integration
* Database Schema Modeler
* Entity Data Modeler
* Full-stack Application Templates
* Debug Perspective