Lujia Zhang
CSSE 477
CM 1405
Paper Review 1

1. Why are extensible architectures important in modern-day applications?

Because extensible architectures allows software developers to productively develop high-quality applications without having to start everything from scratch/reinventing the wheel, rather integrating and building on existing plugins.

2. How are traditional plugins different from pure plugins?

Traditional plugins are not compiled into the host application. They are linked with host application via well-defined interfaces and waiting to be recognized and activated by host application when needed.

In pure plugins, host application is a runtime engine that runs plugins with no inherent end-user functionality. Everything is a plugin. This architecture requires support the extensibility of the plugins by plugins. Each plugin itself becomes a host to other plugins by providing well-defined extension points where other plugins can add functionality.

3. What are extension points and extensions?

Extension points are the hook points that connects host plugins with other plugins.

Extension is when a plug-in contributes an new implementation for an extension point.

4. Enumerate all of the critical services a kernel must support.

- Finding, loading, and running the right plug-in code.
- Maintaining a registry of installed plug-ins and the functions they provide.

- Managing the plug-in extension model and inter-plug- in dependencies.

5. What makes Eclipse a universal pluggable architecture?

Eclipse is a runtime engine that itself is implemented as a number of core plug-ins, except for a tiny bootstrap code.

6. Explain how Eclipse works in a few sentences.

Eclipse starts the core plug-ins to initialize the plug-in registry and the extension model and to resolve plug-in depen- dencies. Other than the core plug-ins, no other plug-in code is run at this time. All the needed plug-in metadata is read from the plug-in manifest files (plugin.xml and/or manifest.mf).

7. Plugin-based applications offer a lot of flexibility for installation. However, this flexibility can also be a source of major headache. Explain how.

Plug-in- based applications may have a higher degree of freedom for installation layout and plug-in discovery. It made it hard to install and find the whole plug-in since only a partcial of the plugin can be access with different privileges.

8. Explain how Eclipse addresses the challenges raised in #7.

Eclipse provides an Update Manager configurator plug-in that picks up plug-ins from the eclipse/plugins folder, as well as from other local plug-in folders linked from the eclipse/links folder or dynamically added when users install new plug-ins to locations of their choice.

9. Explain what difficulties may arise while installing or updating plugins.

Traditional installation issues that arise in any applicationability to roll back changes, migrate existing program data and preferences, or ensure the installation is not corrupted.
Since plug-ins can be originate from various providers that are not related

to each other, it's possible that the resulting configuration has never been tested. So it's not reliable.

10. What are the security implications of pluggable applications?

- Arbitrary plug-ins can be installed from the web, which allows unlimited access to the system.
- Some plug-ins require support for executing custom install code during installation, which allows access to the system.

11. Discuss what the author meant by plugin hell with an example. How does Eclipse approach this problem?

Having multiple versions of plug-in and dependencies to be concurrently installed and exectured at the machine.

Eclipse has adopted a reasonable trade-off convention for concurrent plug-in versions: only versions of plug-ins that contribute code libraries but no plug-in extensions (no user interface contributions, no documentation, and so on) are allowed to coexist in the same runtime instance. For all the other plug-ins, the latest version is usually picked up, unless a configuration file precisely defines what to run.

12. What are some of the approaches to developing scale, pluggable applications?

When designing a plug-in system for scalability, developers must consider various mechanisms that make start-up faster and have a smaller memory footprint. Which requires support for plug-in declarative functionality.

Introduce a packaging and installation component that groups a number of plug-ins to offer a higher level of function.

13. What do features mean to Eclipse ecosystem? Why are they important?

Features are bundles of plug-ins and are considered deployment units with install/update semantics, which will be processed by Eclipse Update Manager.

Features are important because it helps Eclipse to solve the scalability problem when plug-in install/update perform.