ORACLE

Facilitating “Application Specific” aka “Stripped” Implementations

JCP EC Discussion
July 8, 2014

DRAFT – FOR DISCUSSION PURPOSES

The following is intended to outline our general product direction. It is intended for information purposes only, and may not be incorporated into any contract. It is not a commitment to deliver any material, code, or functionality, and should not be relied upon in making purchasing decisions.

The development, release, and timing of any features or functionality described for Oracle′s products remains at the sole discretion of Oracle.

©2014 Oracle Corporation

DRAFT – FOR DISCUSSION PURPOSES

Goal:

- Allow unused elements (e.g., methods, classes or even whole packages) to be removed or ‘stripped’ from a TCK-compliant implementation (e.g., Java SE and Java ME, but other specifications if desired), to reduce storage and memory consumption.

What are “Application Specific” aka “Stripped” Implementations?

DRAFT – FOR DISCUSSION PURPOSES

“An implementation based upon a complete and TCK-compliant (e.g., Java SE or Java ME) implementation, but distributed with a dependent application that uses the implementation in a closed environment where unused elements are removed, or "*stripped*“, in order to reduce storage and memory consumption.

DRAFT – FOR DISCUSSION PURPOSES

Application Specific Implementation - Basics

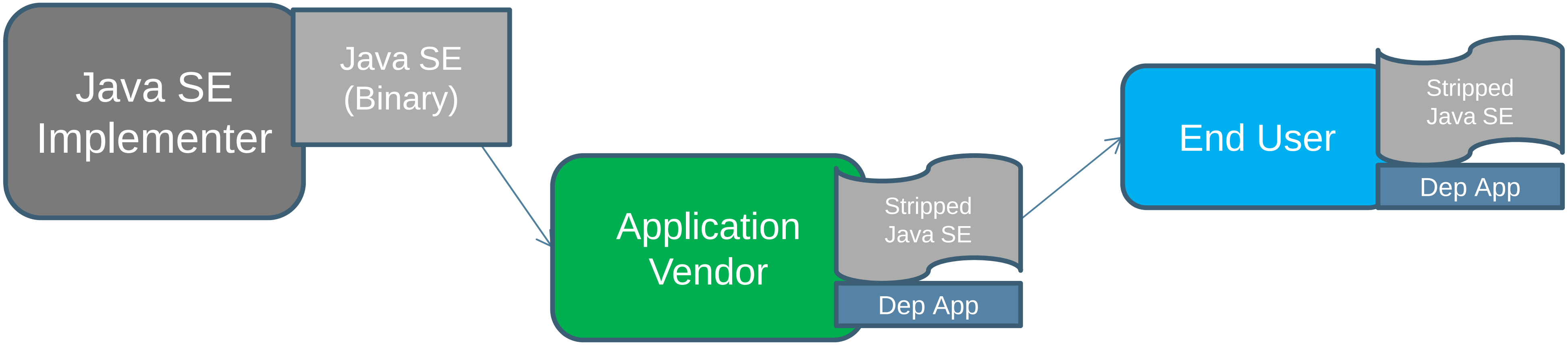
AKA “Stripped Implementation”

- Based on complete and TCK Spec compliant implementation
- Distributed only with a Dependent Application
- Unused elements may be removed, or ‘stripped’ to reduce storage and memory consumption
  - E.g., methods, classes or even whole packages
  - Manually, via provided tools, automated on deployment, etc.

DRAFT – FOR DISCUSSION PURPOSES

Application Specific Implementation

Ex 1: ‘Stripping’ and Redistribution of Java SE by Application Developers

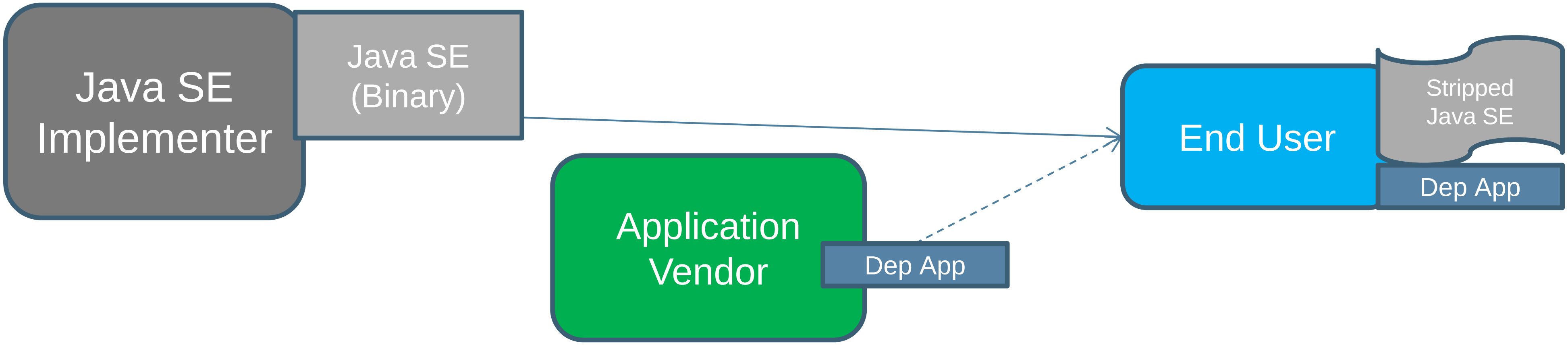


- An Application Developer licenses Java SE from an Implementer, “strips” it with their dependent application and redistributes it further
- Either Java SE or Java ME (eventually other JSRs)

DRAFT – FOR DISCUSSION PURPOSES

Application Specific Implementation

Ex 2: ‘Stripping’ of Java SE by an End User



- An End User either builds their own dependent application, or licenses one from an Application Vendor, and then ‘strips’ an implementation provided by a Java SE Implementer
- Either Java SE or Java ME (eventually other JSRs)

Sounds great, ship it!

DRAFT – FOR DISCUSSION PURPOSES

Additional Constraints

Protecting Compatibility

Application Specific Implementations must:

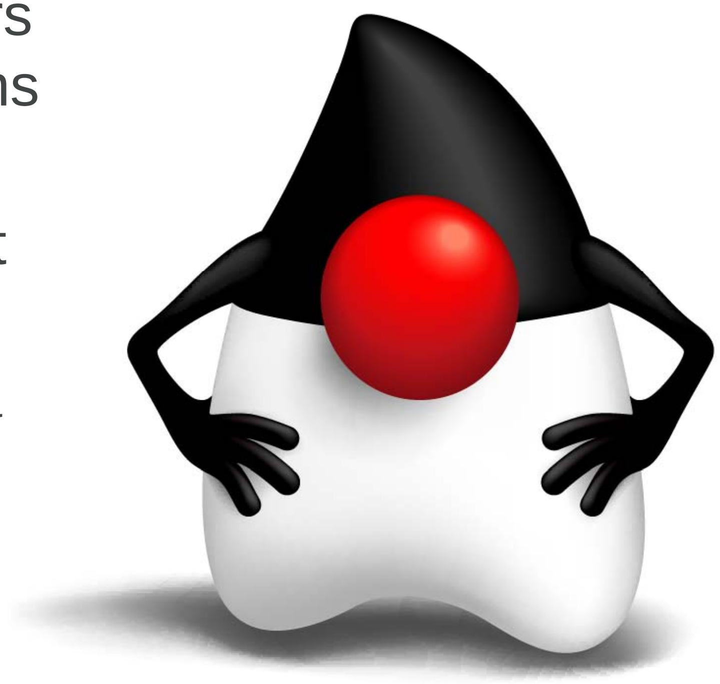
- Be restricted from further stripping or other modifications downstream once created
- Function identically to the ‘non-stripped’ Full Implementation
- Be “Closed” in that they do not expose APIs and cannot execute code other than the dependent application
  - To prevent sub-setting of Java just for the sake of it. Application developers should always start from Full Implementation.

Requires changes to Licensing and Specs

DRAFT – FOR DISCUSSION PURPOSES

Licensing Proposal

- Make completely optional for Implementers to allow “Stripping” of their implementations
- Require the “Stripper” to enter agreement with Spec Lead, and pass an optional part of TCK
  - Application Developer, End User or even a Java Implementer
- Create an enforceable relationship with Spec Lead

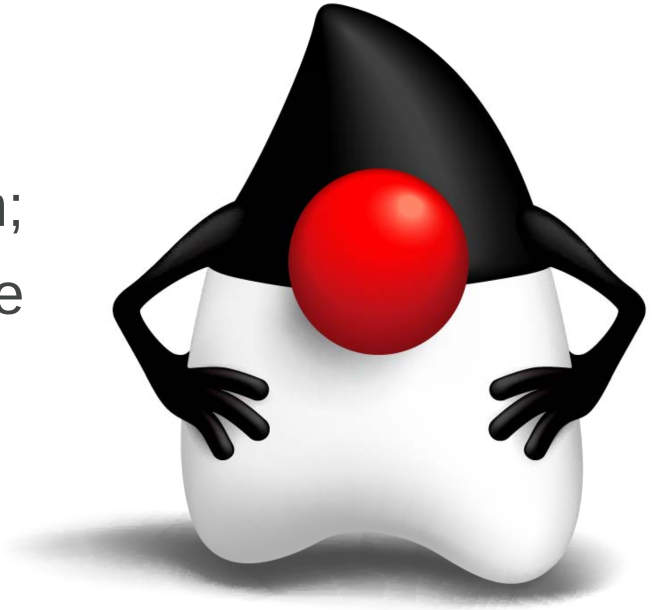


DRAFT – FOR DISCUSSION PURPOSES

Optional Part of TCK

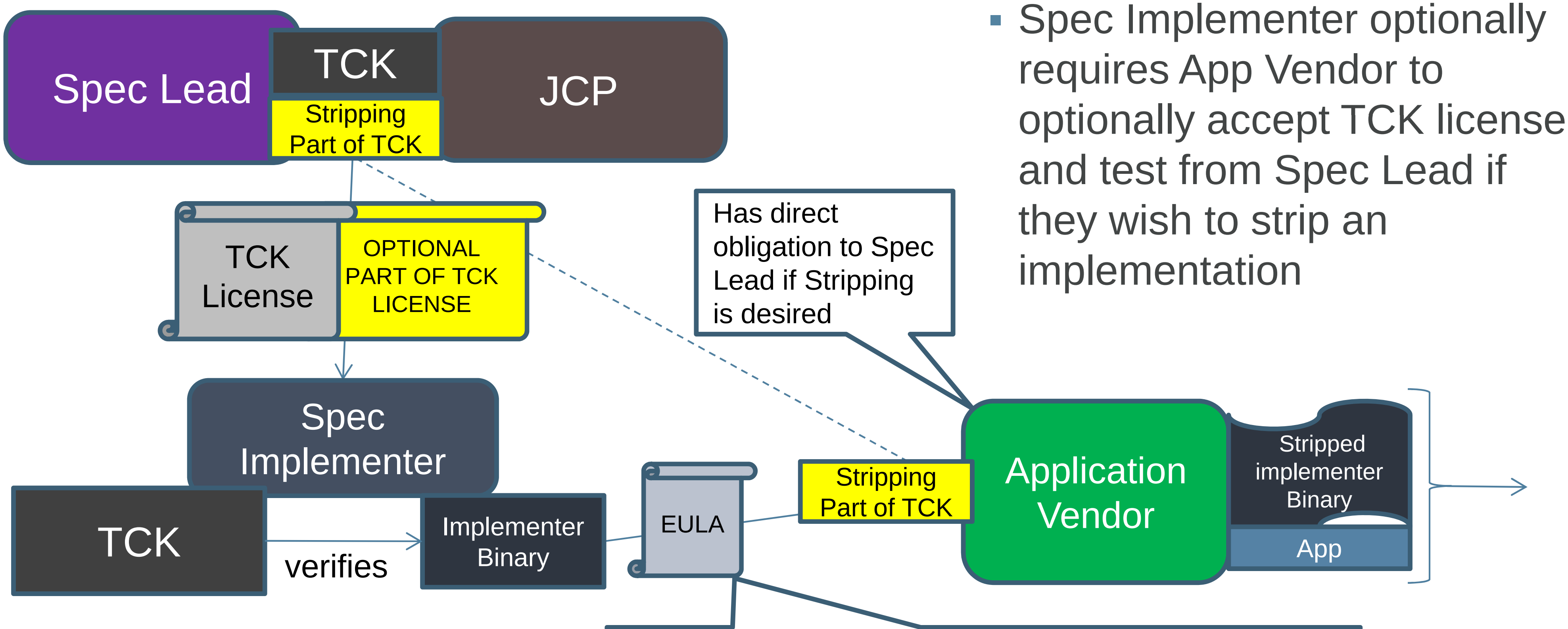
Example “tests” in the optional part of the TCK

- Your stripped implementation is:
  - Derived from a complete, conventionally compatible implementation of the platform;
  - Does not expose APIs and cannot execute code other than the included Application;
  - Functions identically to how it functions with the Full implementation.
- May just be a ‘checklist’ vs provided software test suite



DRAFT – FOR DISCUSSION PURPOSES

Licensing Proposal – App Vendor POV



Has direct obligation to Spec Lead if Stripping is desired

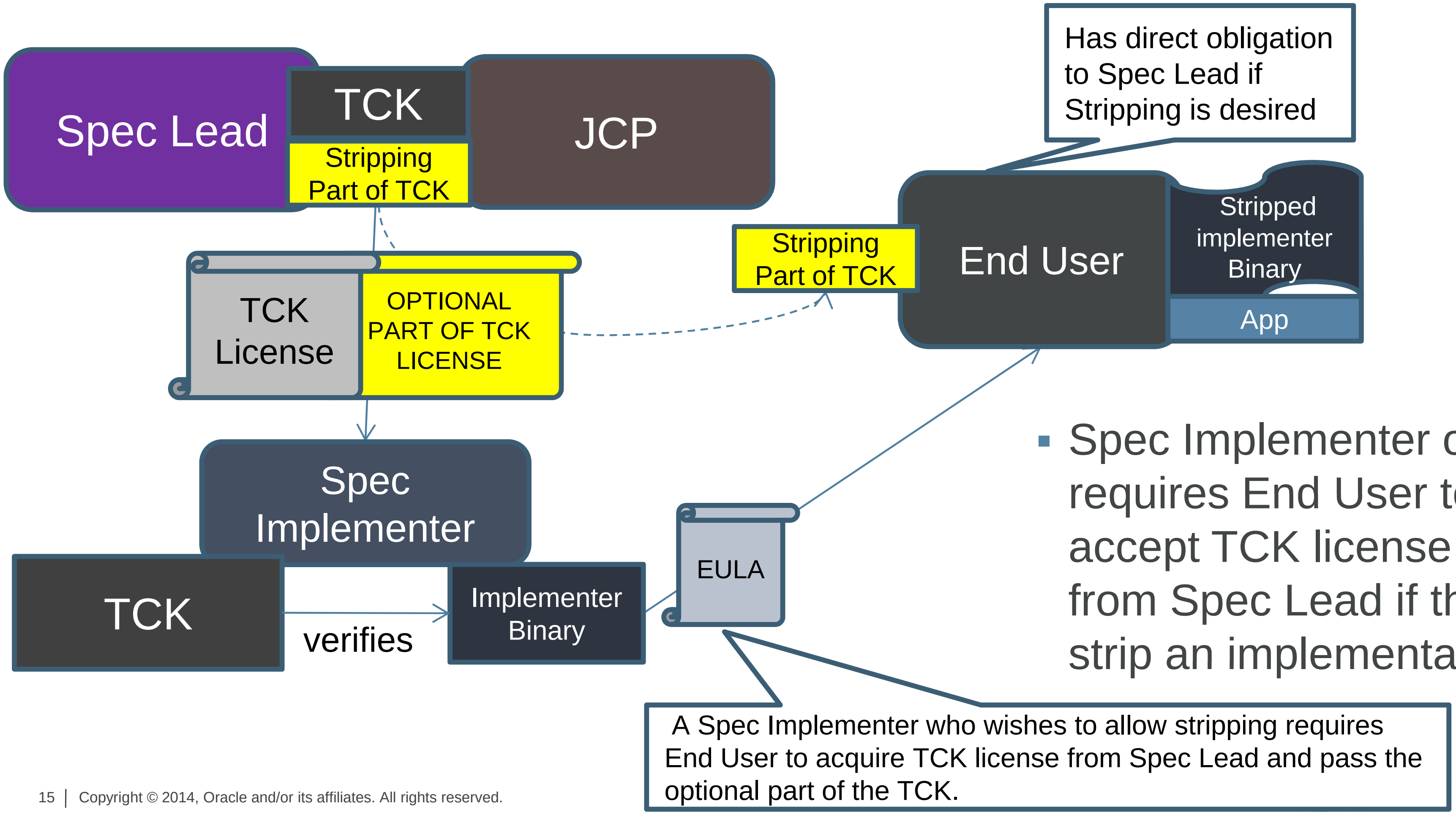
A Spec Implementer who wishes to allow stripping requires App Vendor to acquire TCK license from Spec Lead and pass the optional part of the TCK.

- Spec Implementer optionally requires App Vendor to optionally accept TCK license and test from Spec Lead if they wish to strip an implementation

ORACLE

DRAFT – FOR DISCUSSION PURPOSES

Licensing Proposal – End User POV



Has direct obligation to Spec Lead if Stripping is desired

A Spec Implementer who wishes to allow stripping requires End User to acquire TCK license from Spec Lead and pass the optional part of the TCK.

- Spec Implementer optionally requires End User to optionally accept TCK license and test from Spec Lead if they wish to strip an implementation

DRAFT – FOR DISCUSSION PURPOSES

Summary of Impact on Relevant Documents (1 of 2)

- JSPA – No changes required
- Specification License – No changes required
- Specification:
  - Define “Fully Implemented” and “Application Specific”
  - Add condition that, once stripped, implementations become “closed” (no further changes, no exposed APIs, etc)

DRAFT – FOR DISCUSSION PURPOSES

Summary of Impact on Relevant Documents (2 of 2)

- TCK License
  - Creation of the “Optional part of TCK License” related to stripping
  - Updates to allow downstream “stripping” upon condition of accepting Spec Lead’s “Optional part of TCK License”
- TCK
  - Addition of “Optional part” of TCK related to stripping
- Implementer’s Binary License (e.g., the “BCL” for Oracle Implementations)
  - Updates to allow direct licensee “stripping” upon condition of accepting Spec Lead’s “Optional part of TCK License”

Java™