CN4J
Jakarta EE & MicroProfile
Relation

2023-03-06 Jan Westerkamp

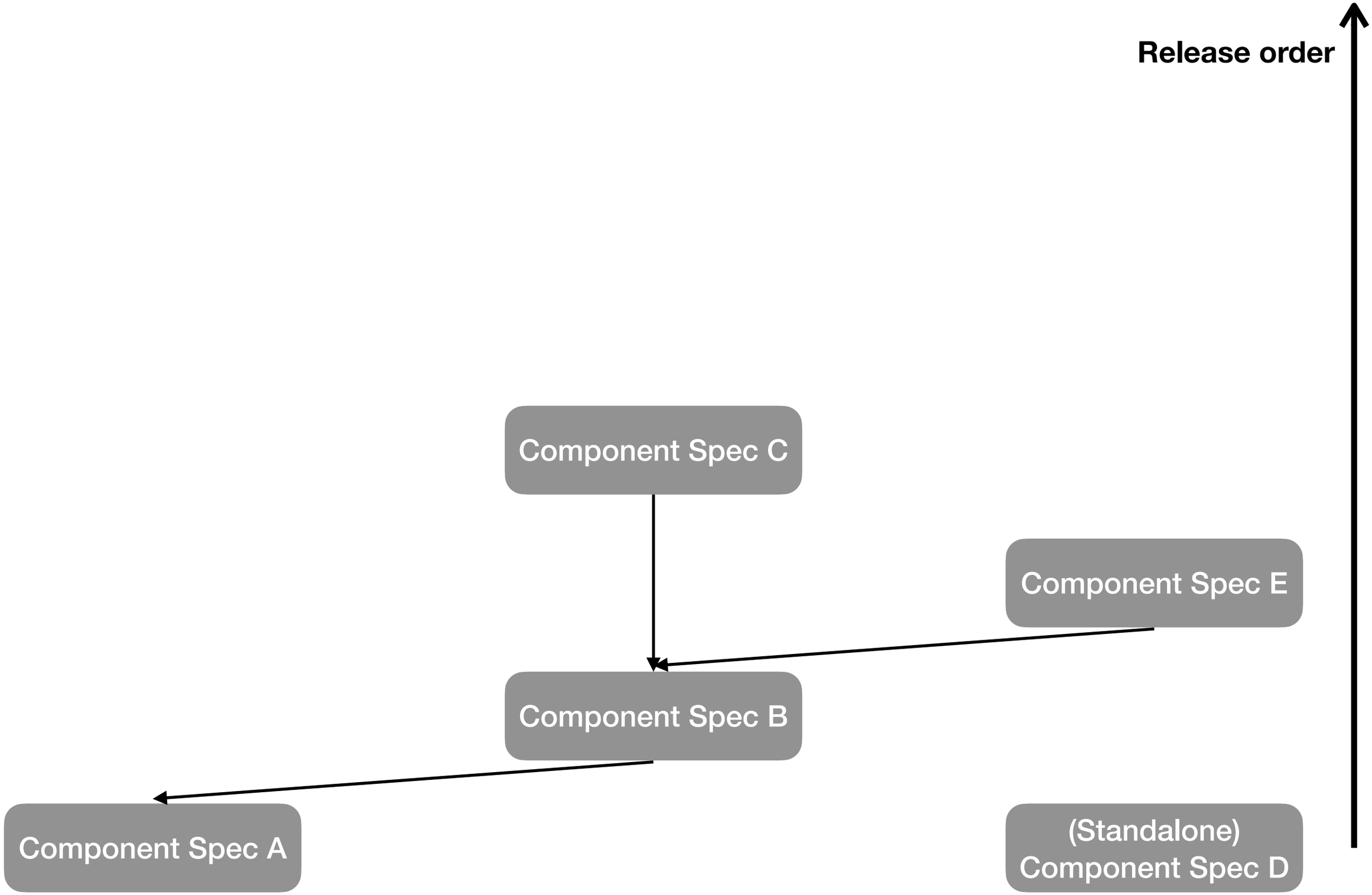
Agenda

- Organisations
- Component Specs
- Umbrella Specs (Profiles/Platform)
- Spec Implementations
- System Environment
- Jakarta EE & MicroProfile
- CN4J Roadmap
- Issues & Solutions
    - Continuous Integration
    - MicroProfile Parent
    - MicroProfile Spec
    - Jakarta EE Parent
    - Jakarta EE Spec
    - Jakarta EE Platform
    - MicroProfile Metrics
    - Jakarta Security & MP JWT
    - MP Config & Jakarta Config
    - MP Telemetry & OpenTelemetry
    - MP Metrics & MP Telemetry
    - MP REST Client & Jakarta REST
- Security Issues TBD
- Conclusions TBD

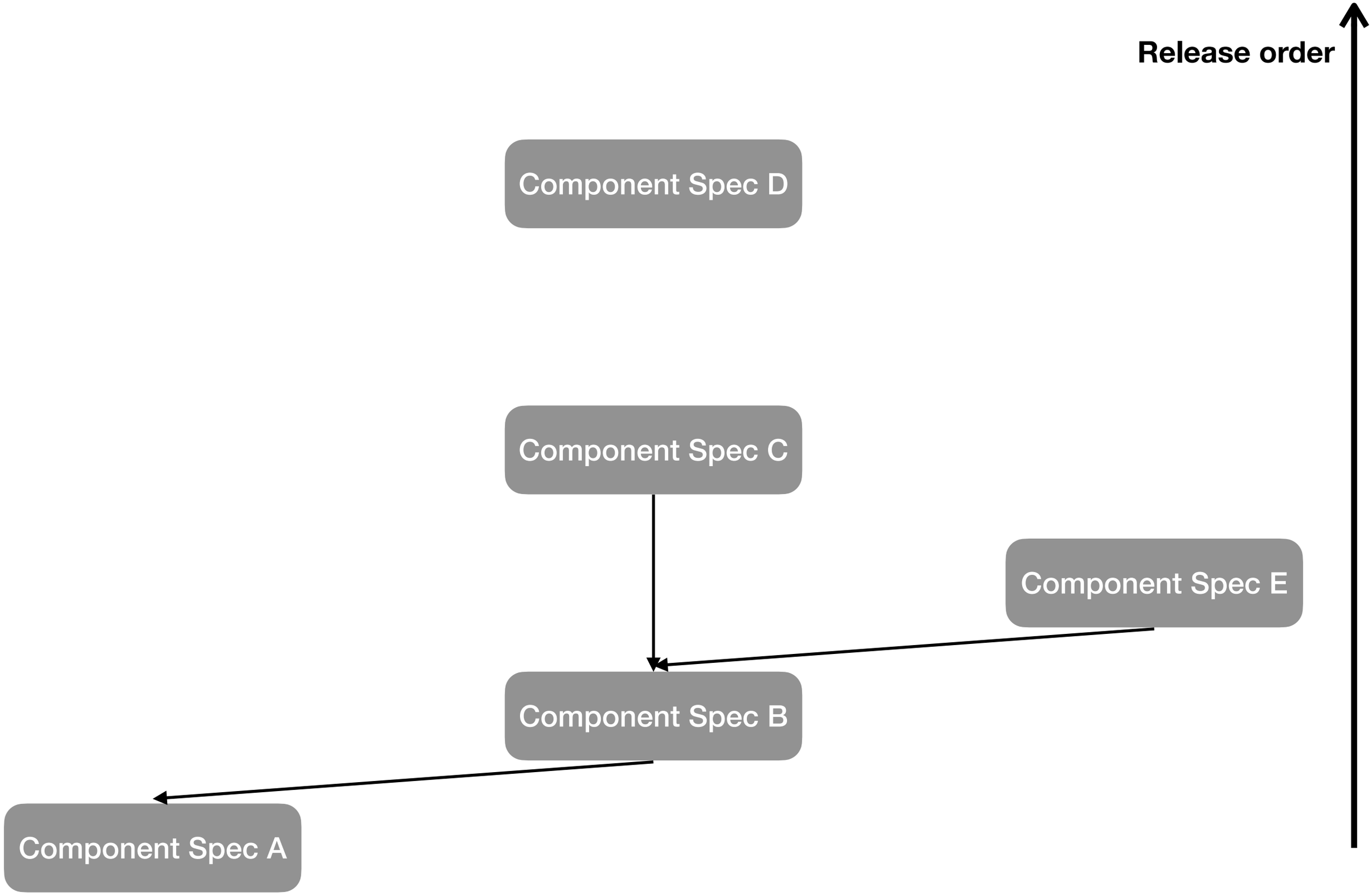
Organisations

- Eclipse Foundation
  - Jakarta EE Working Group
  - MicroProfile Working Group
  - CN4J Association
- Broadcom
  - VMware
    - VMware Tanzu (Pivotal)
- Other (implementation providers, external dependency providers, application development organisations)

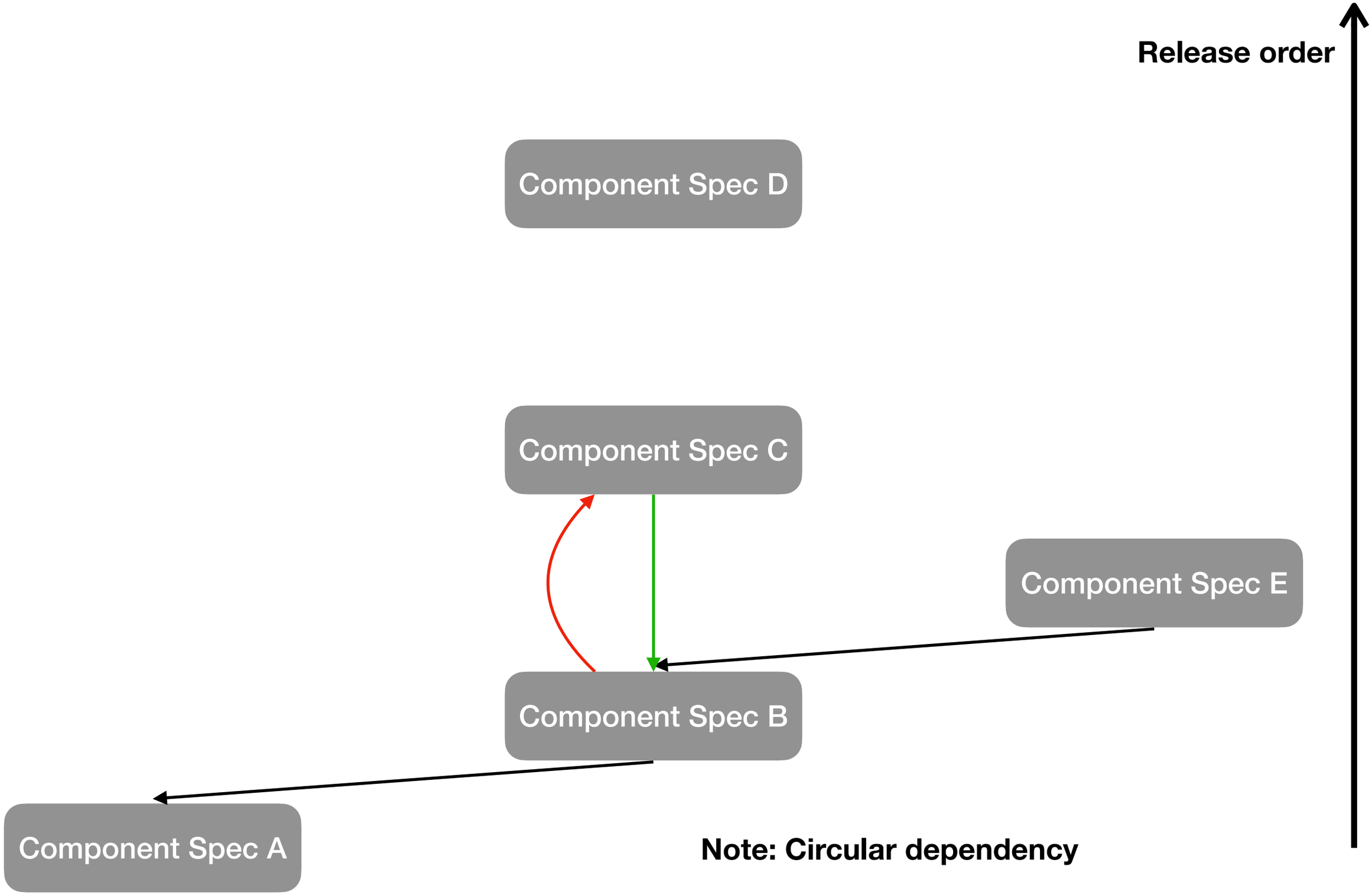
Component Specs



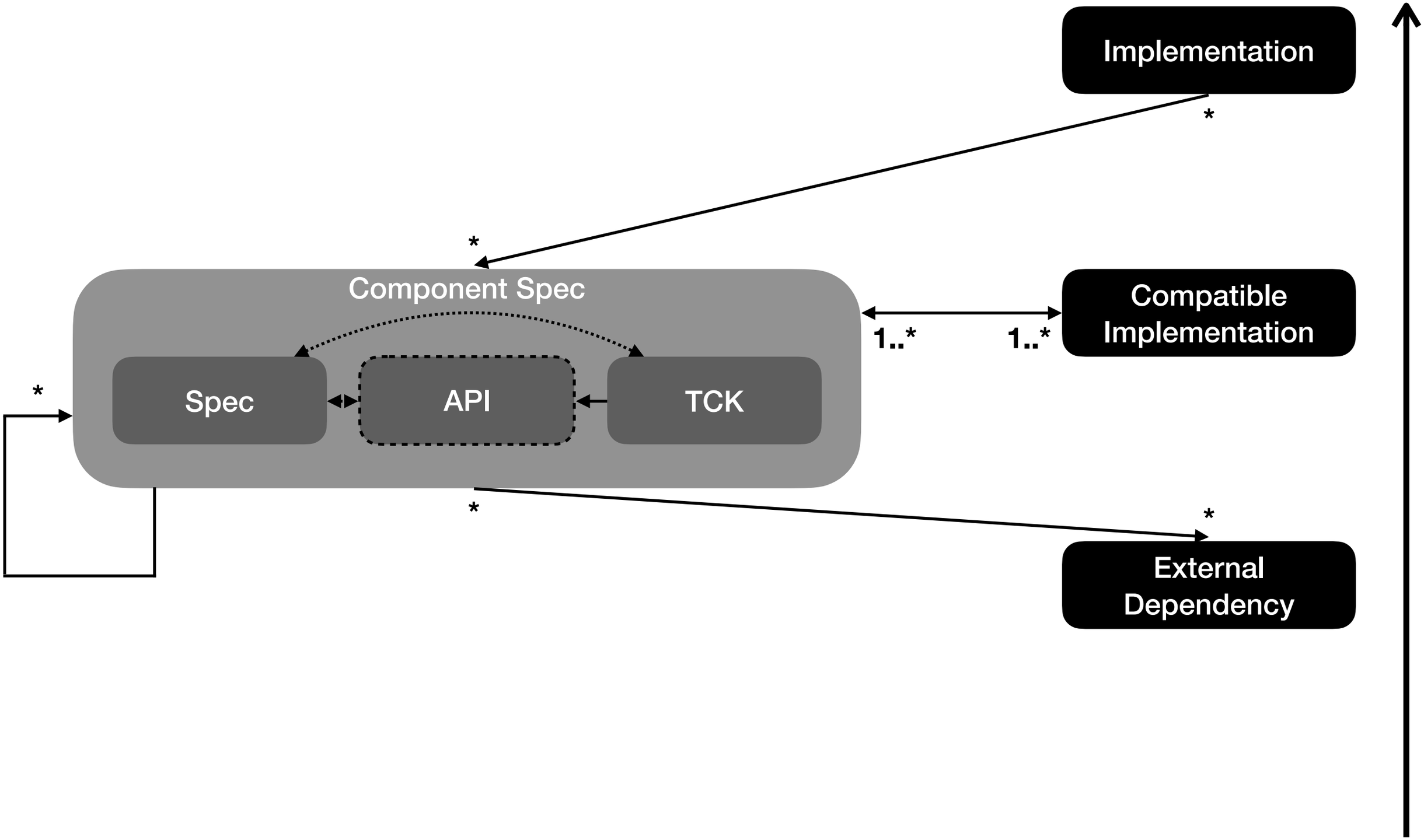
Component Specs



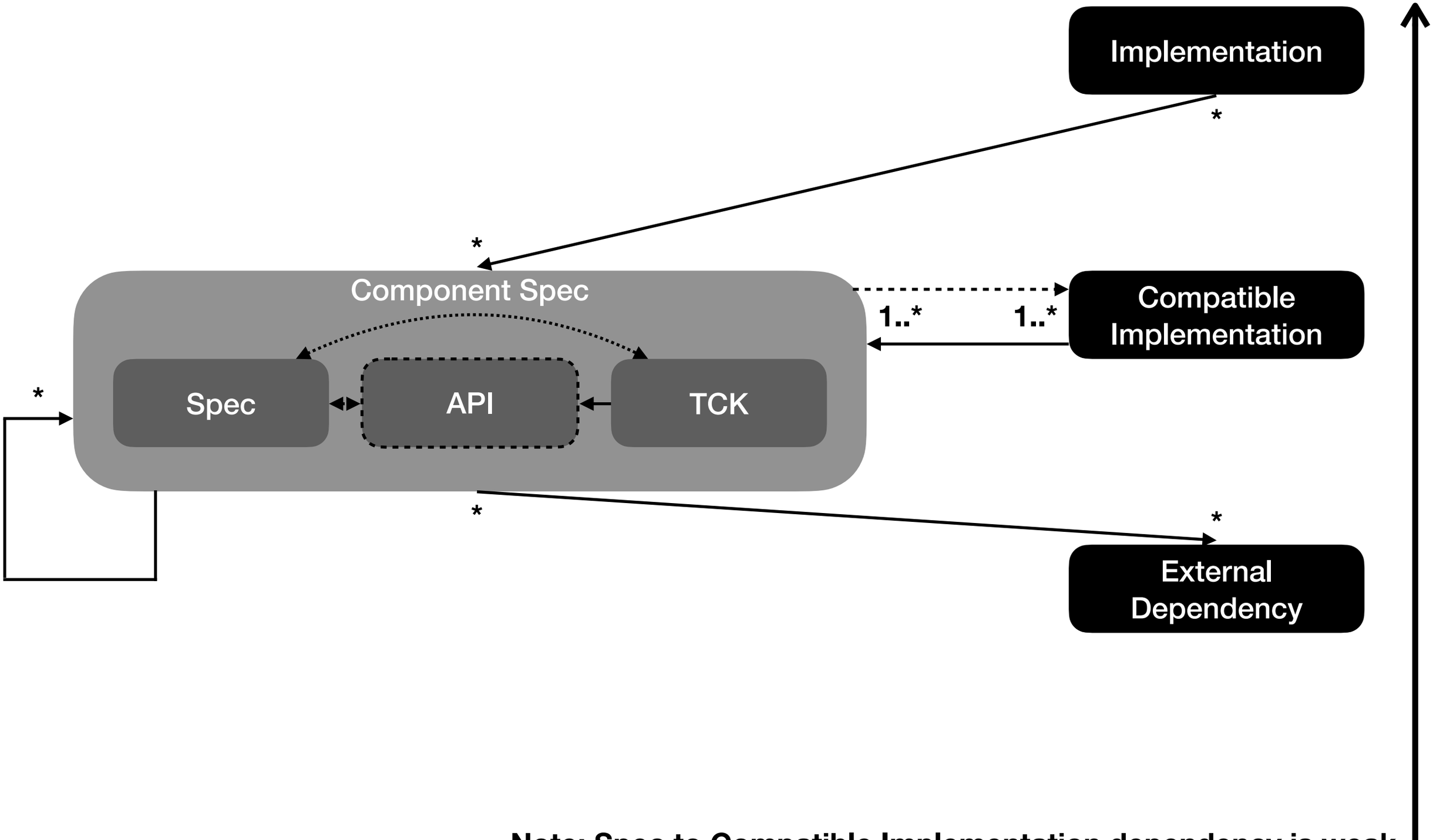
Component Specs



Component Spec Detail

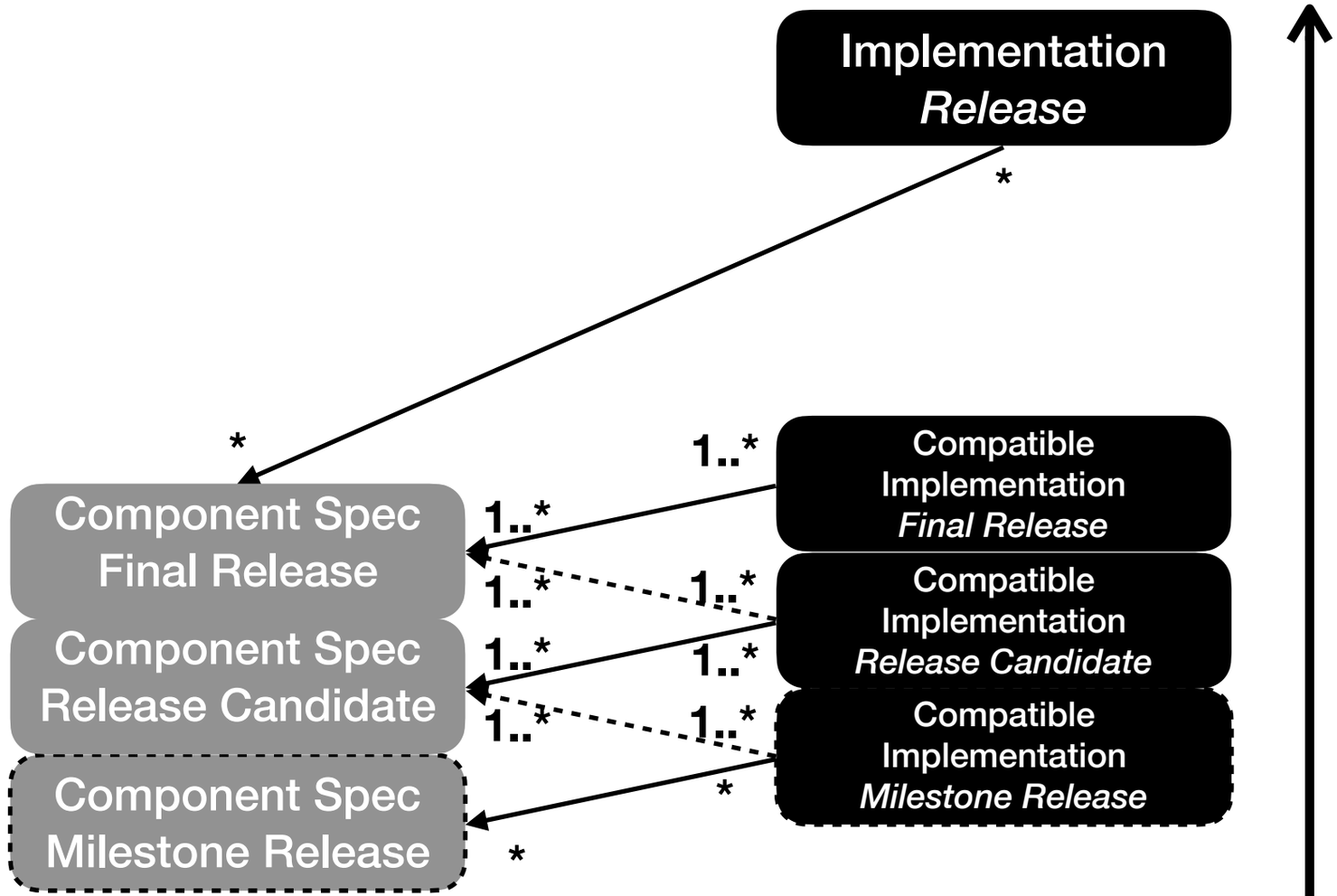


Component Spec Detail



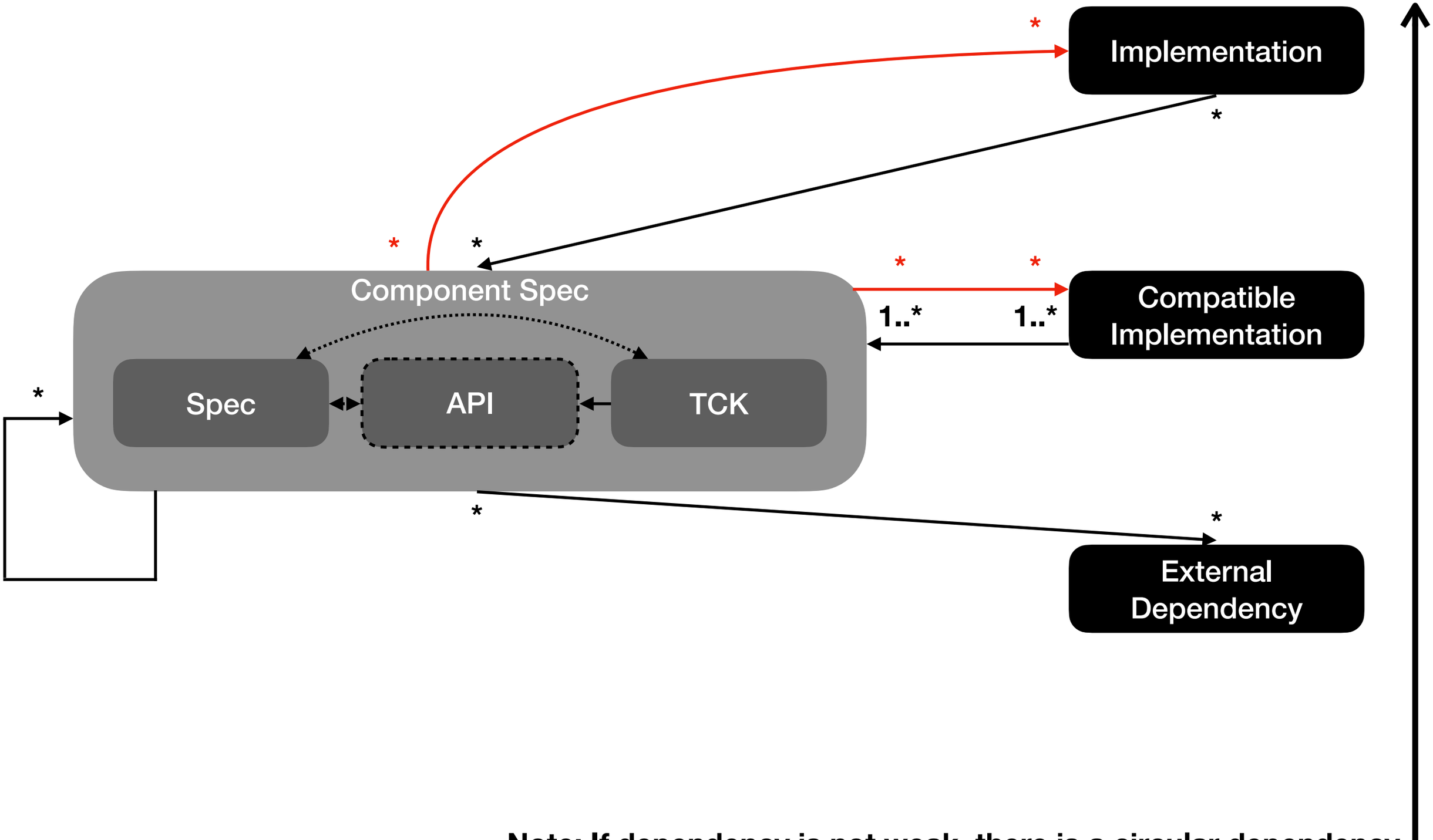
Note: Spec to Compatible Implementation dependency is weak

Component Spec Detail



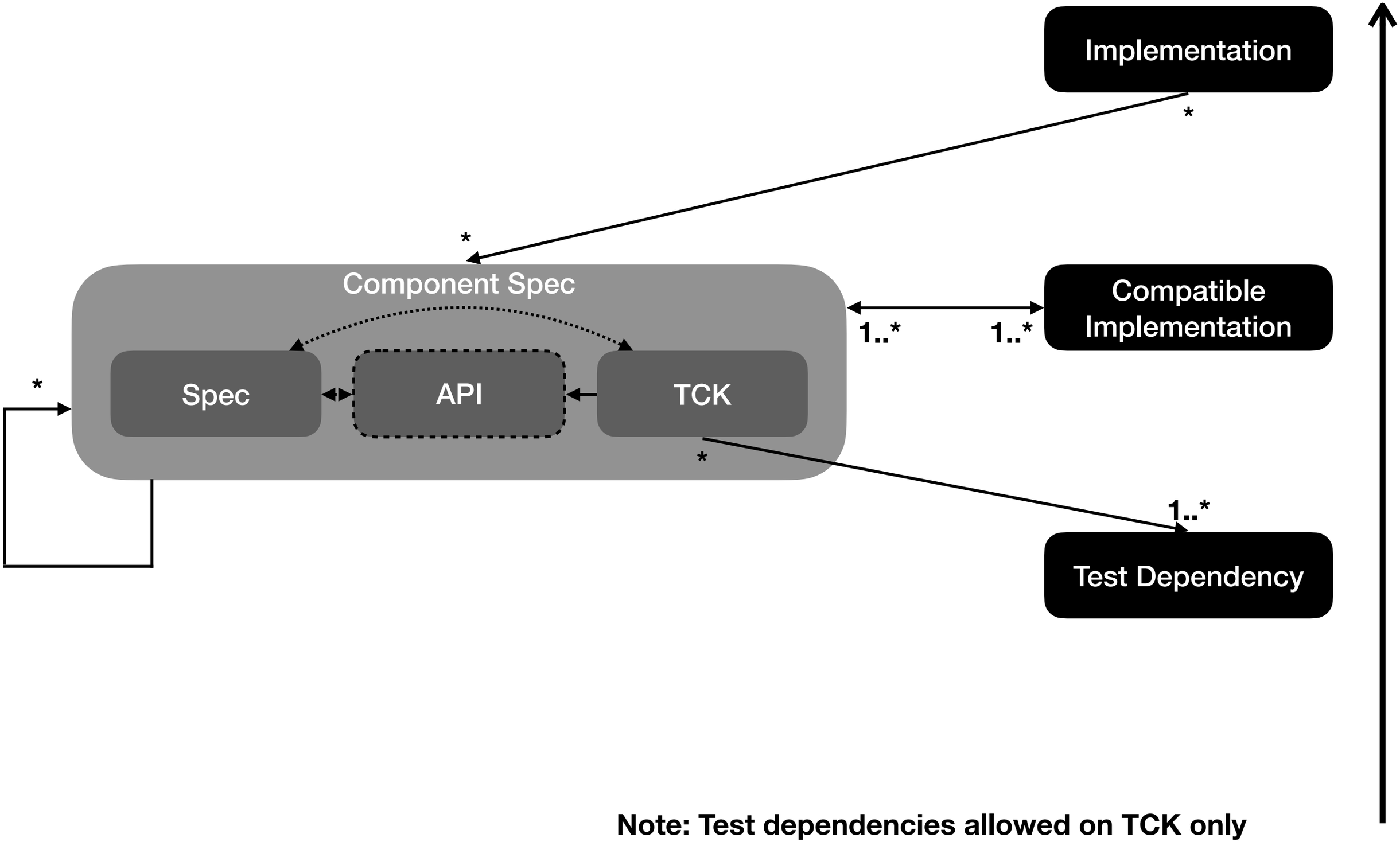
Note: Implementation projects are free in choosing their own version schema

Component Spec Detail



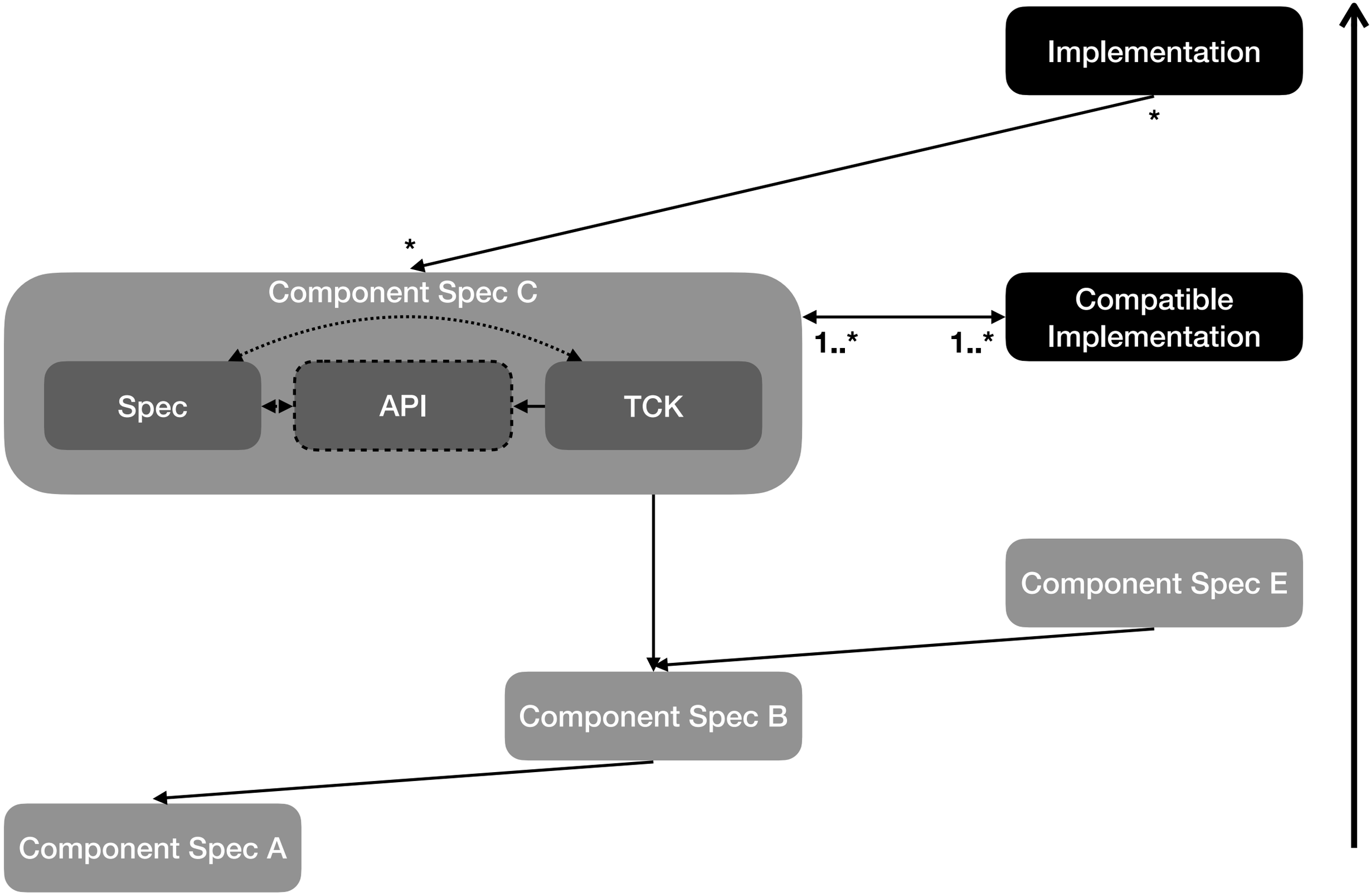
Note: If dependency is not weak, there is a circular dependency

Component Spec Detail

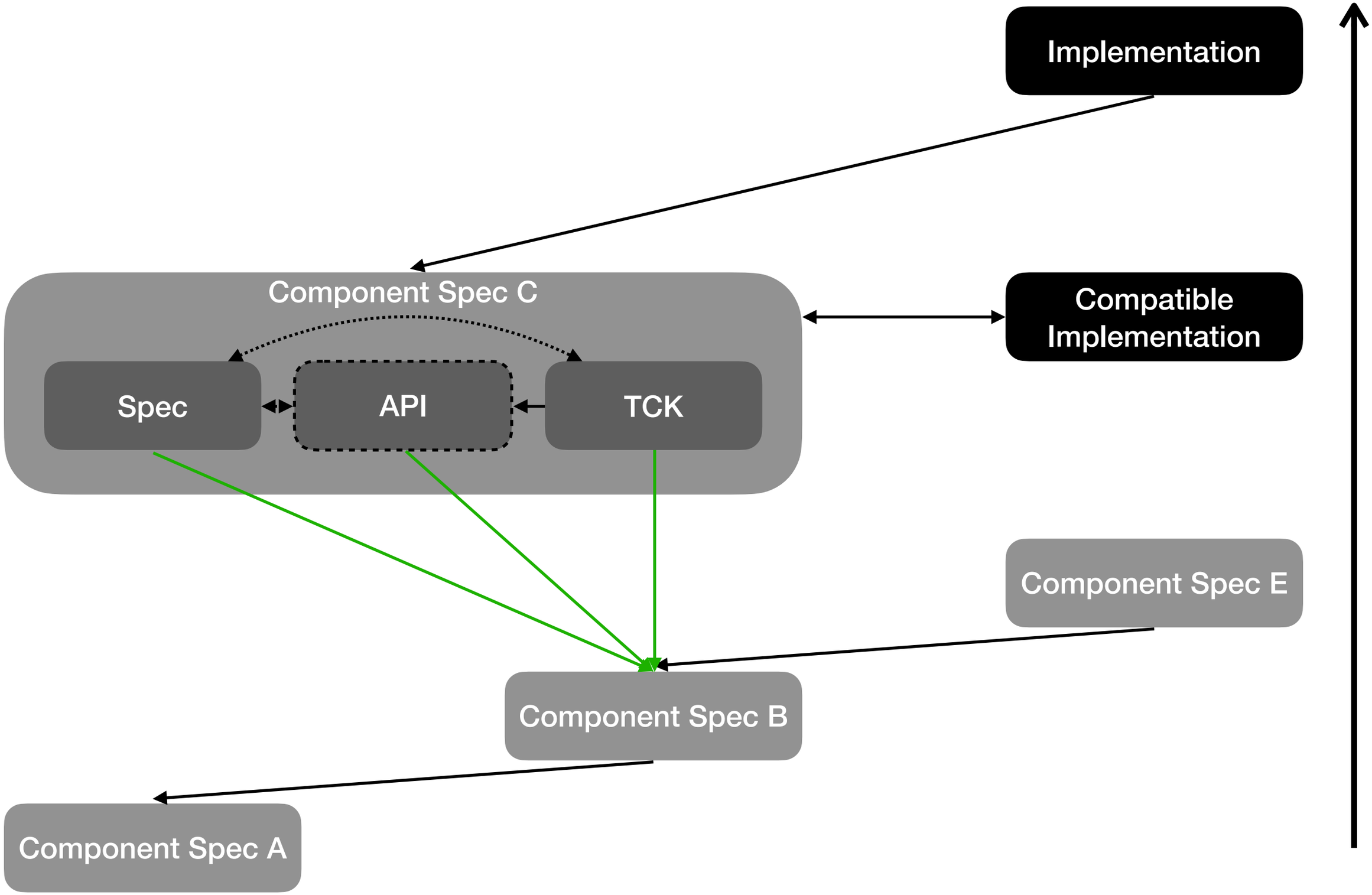


Note: Test dependencies allowed on TCK only

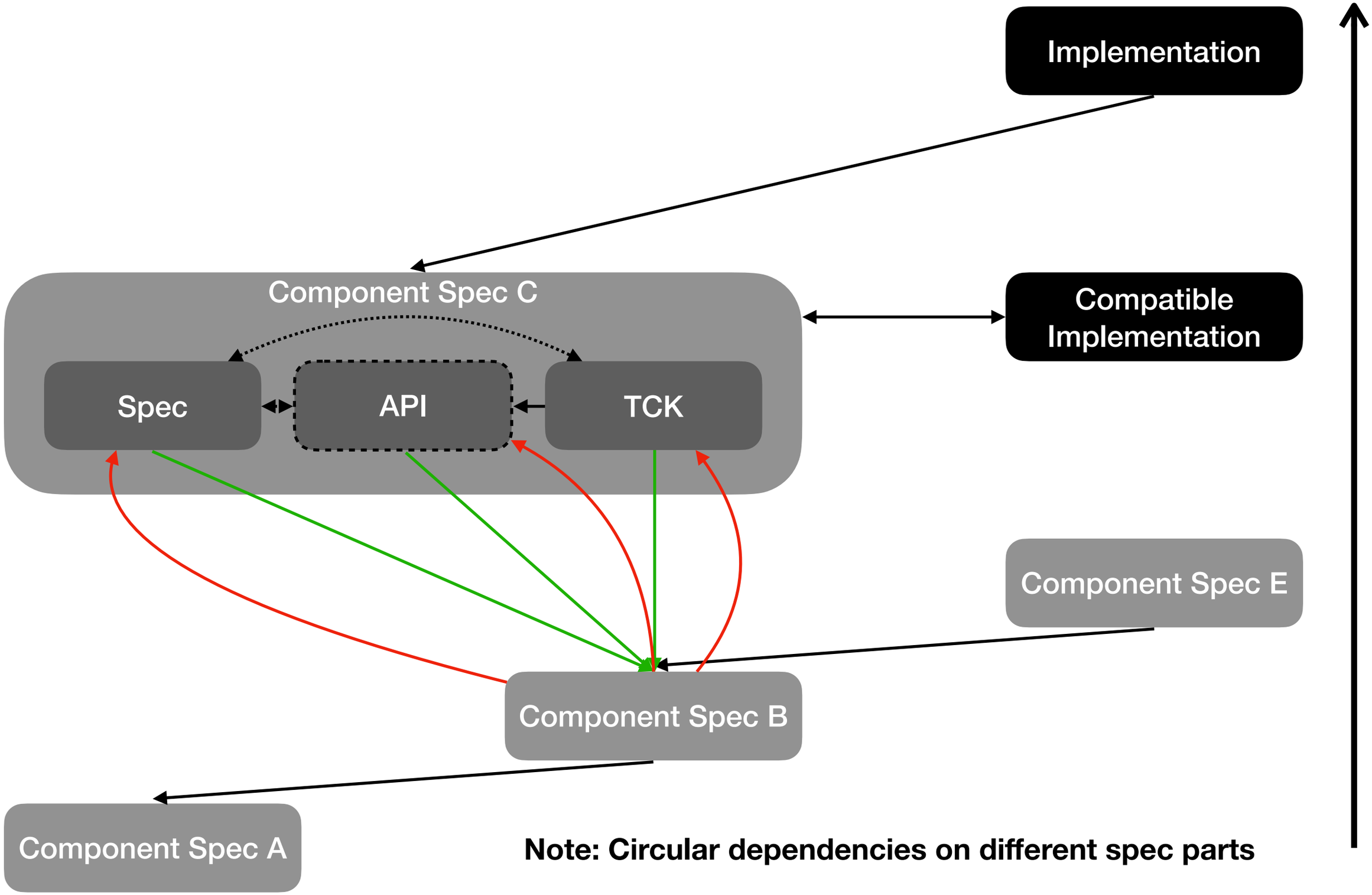
Component Spec Detail



Component Spec Detail

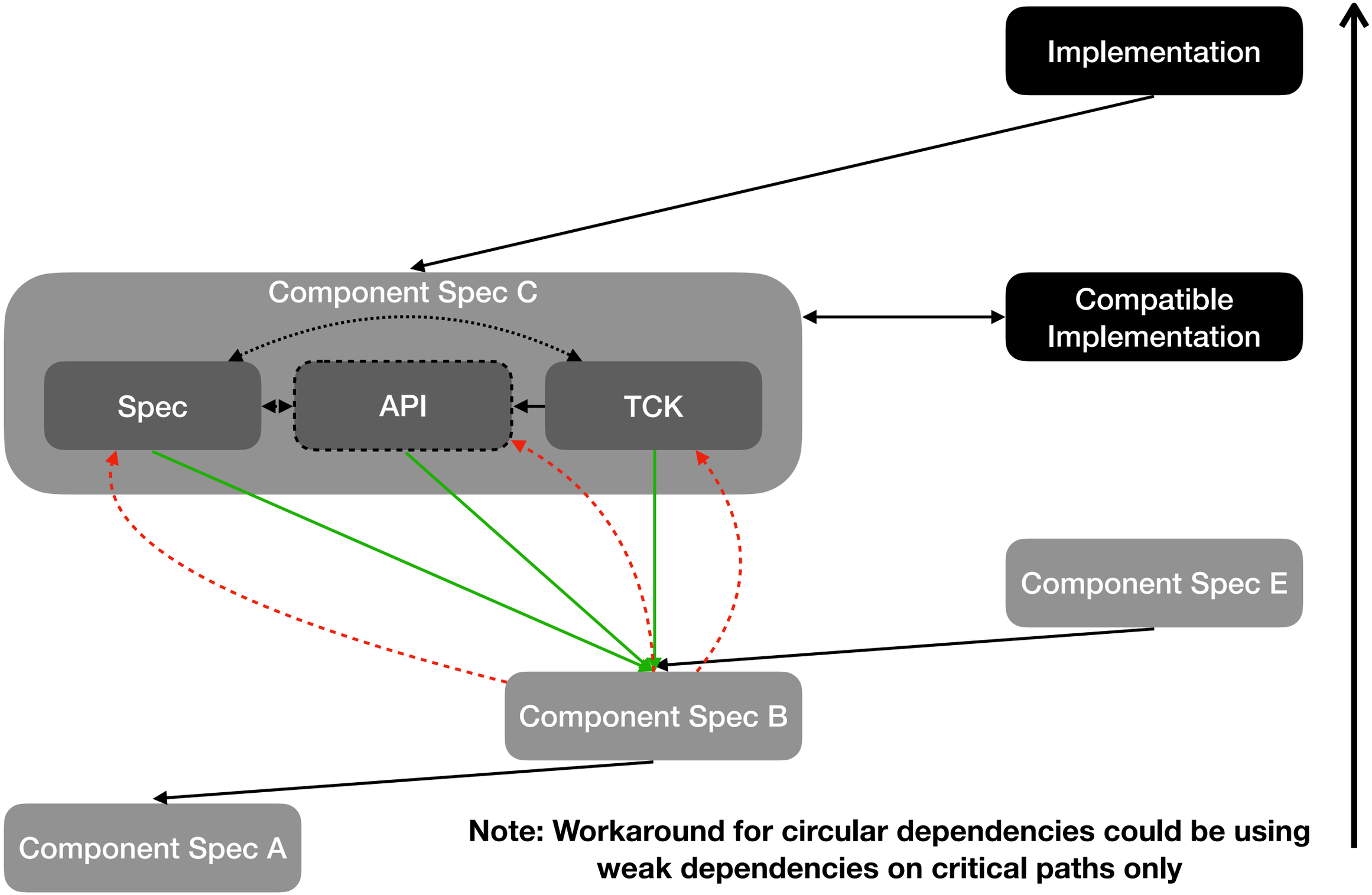


Component Spec Detail



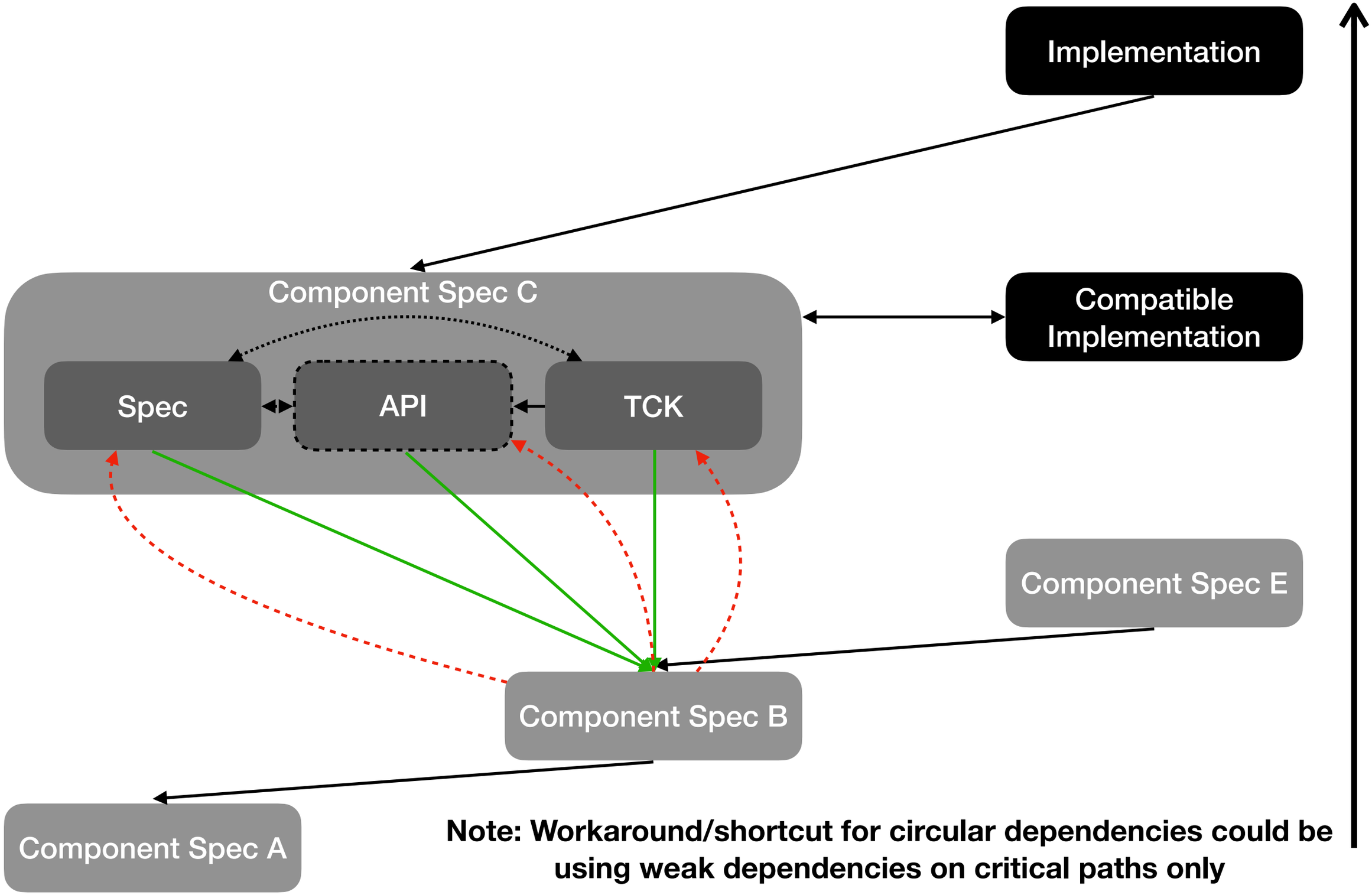
Note: Circular dependencies on different spec parts

Component Spec Detail



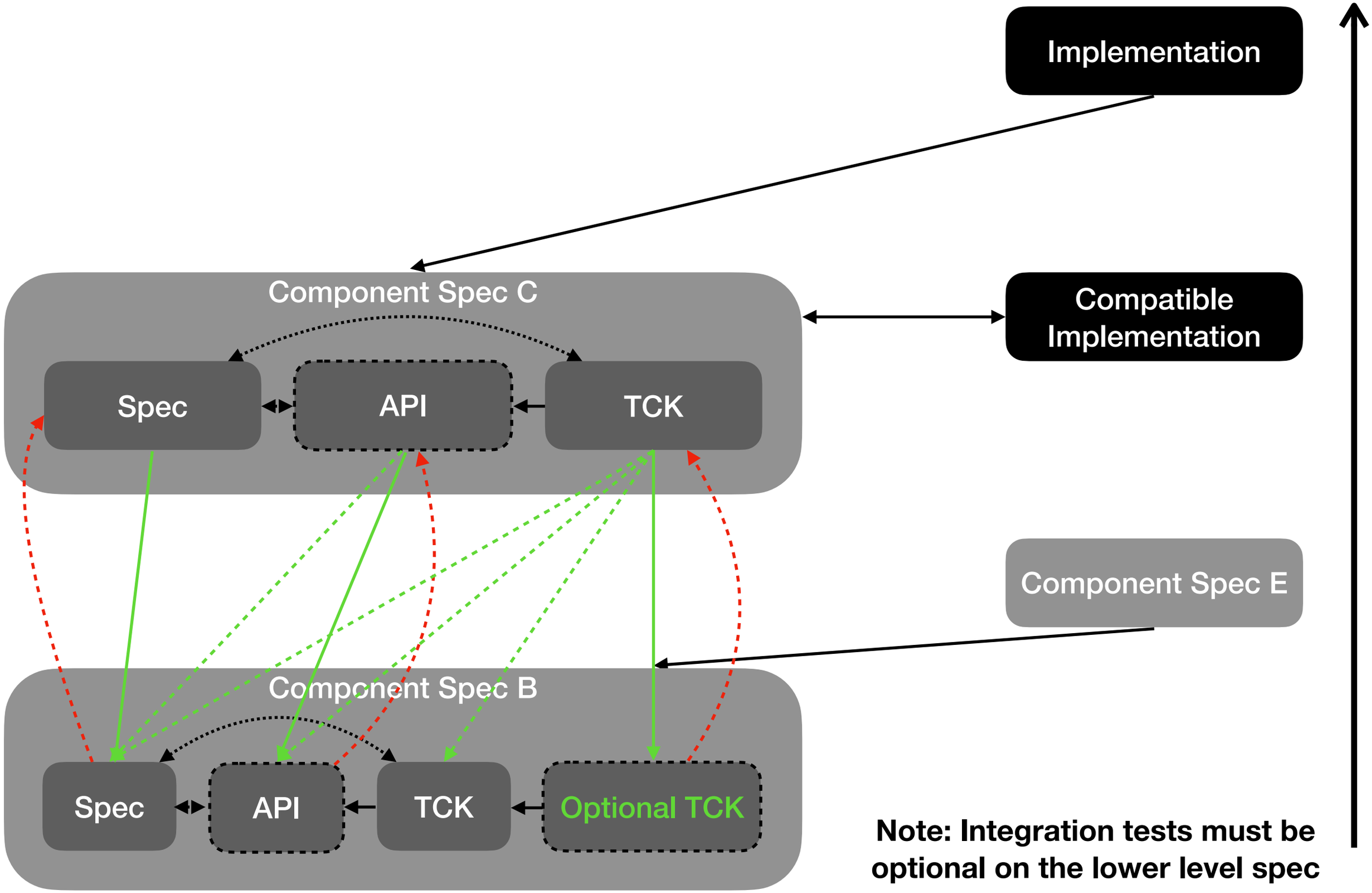
Note: Workaround for circular dependencies could be using weak dependencies on critical paths only

Component Spec Detail

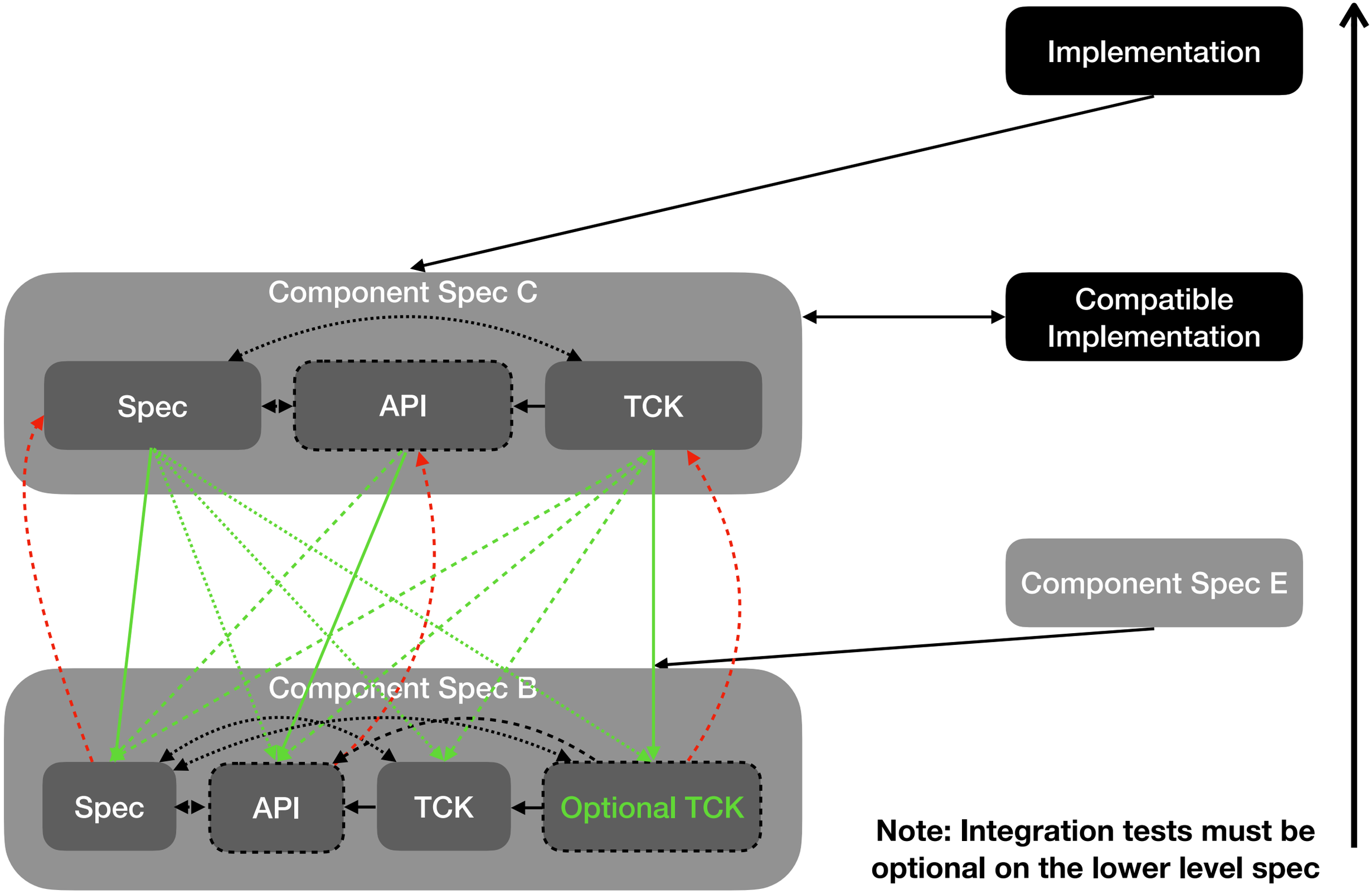


Note: Workaround/shortcut for circular dependencies could be using weak dependencies on critical paths only

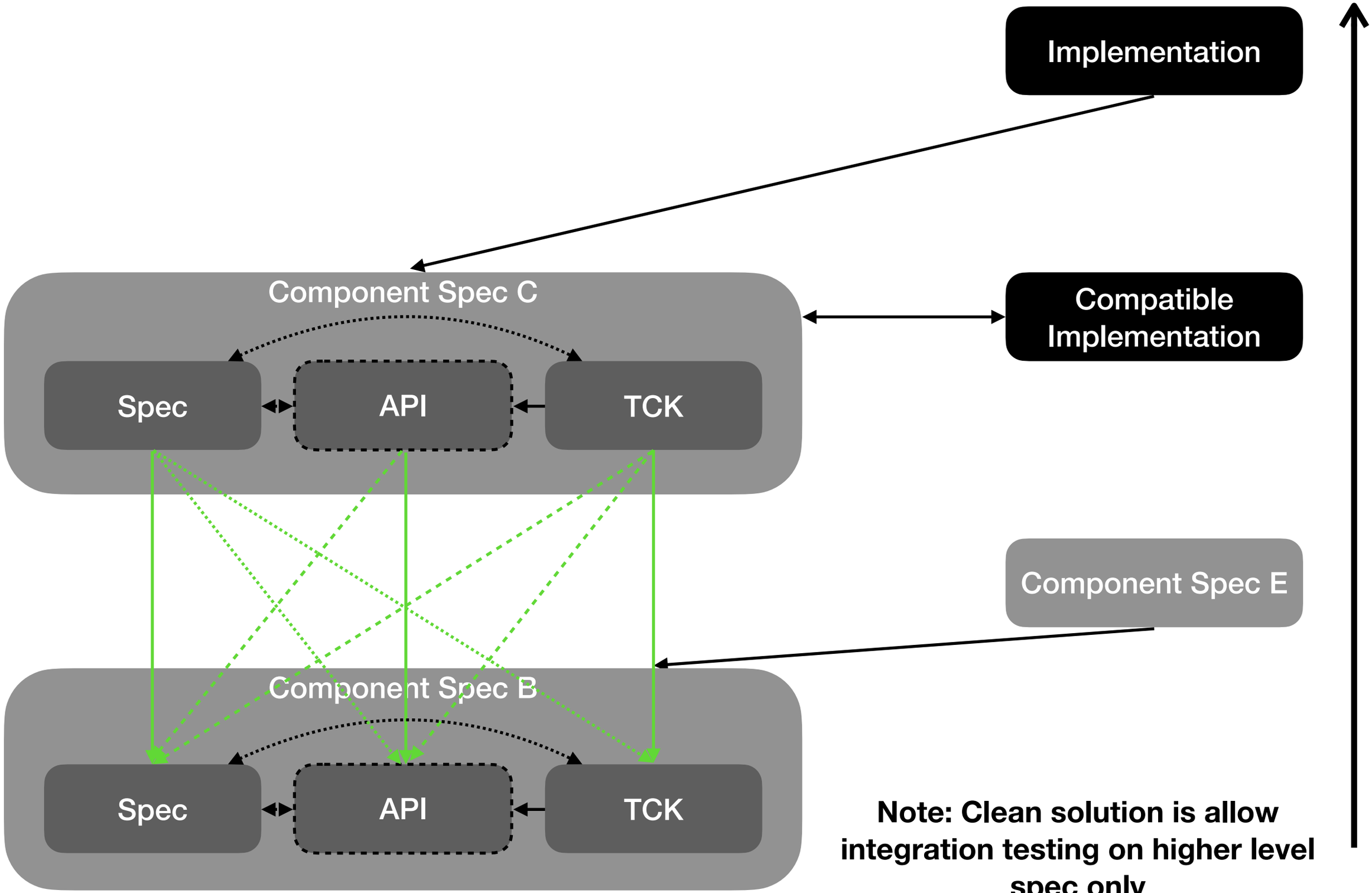
Component Spec Detail



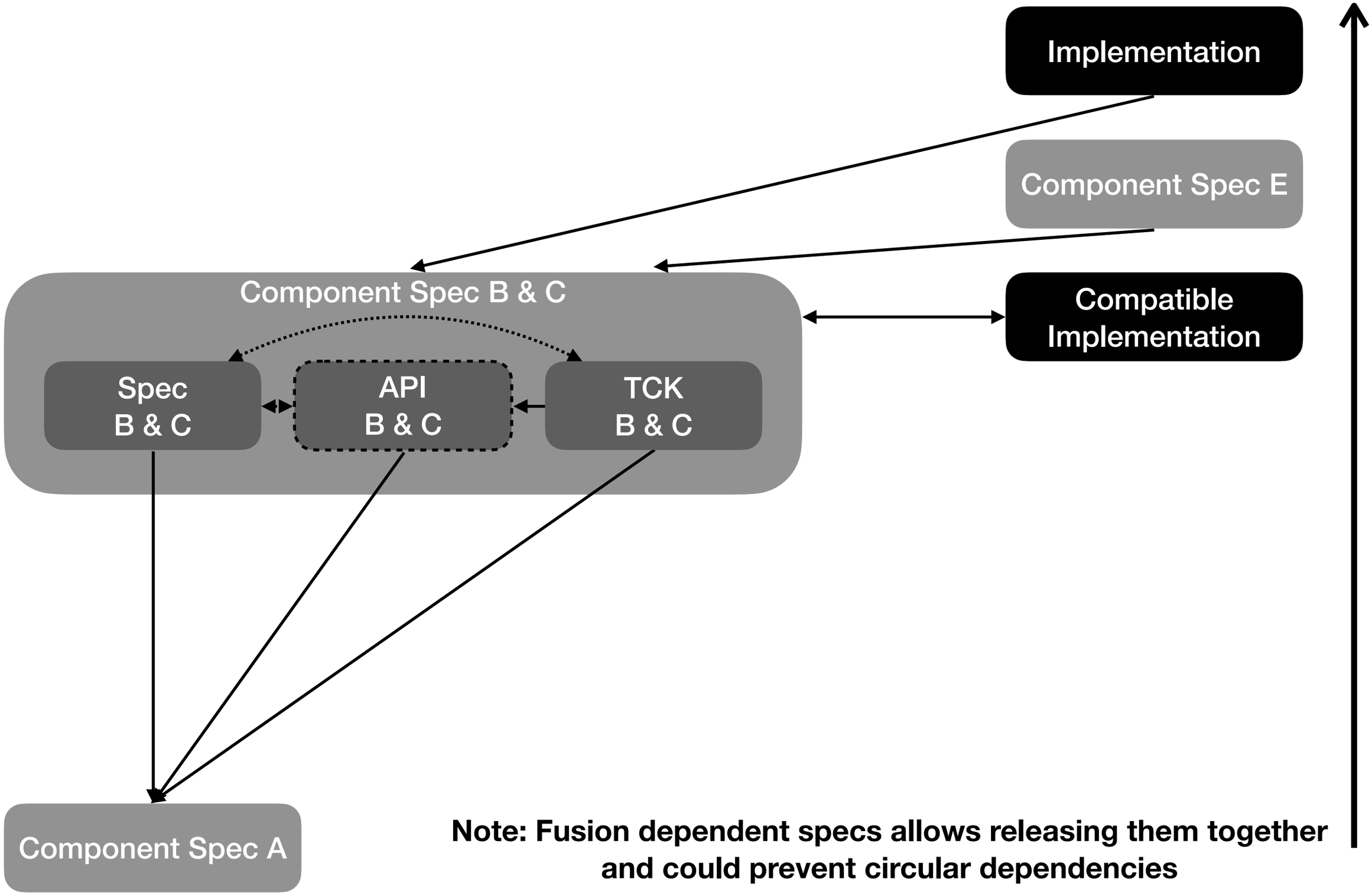
Component Spec Detail



Component Spec Detail

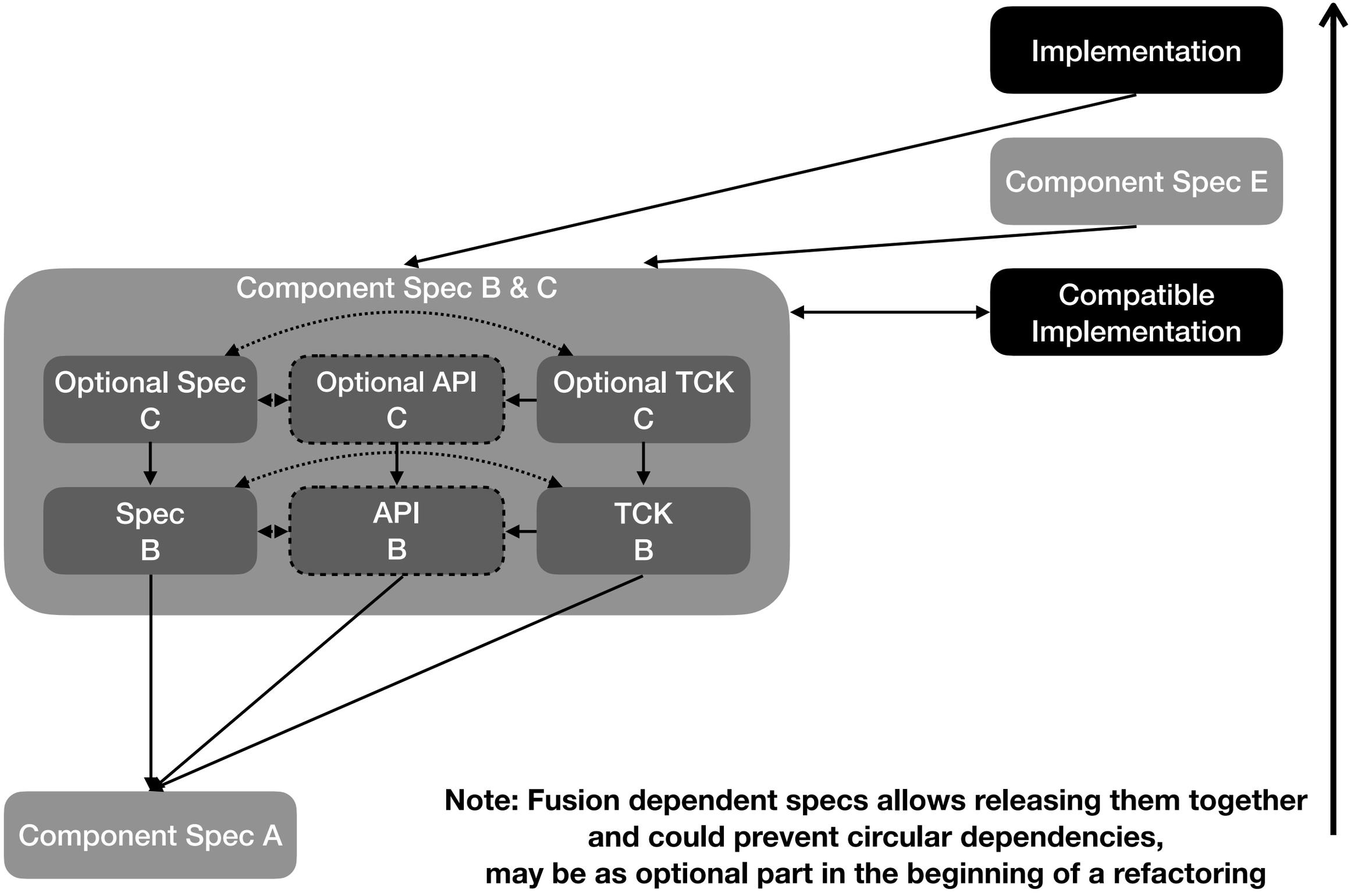


Component Spec Detail



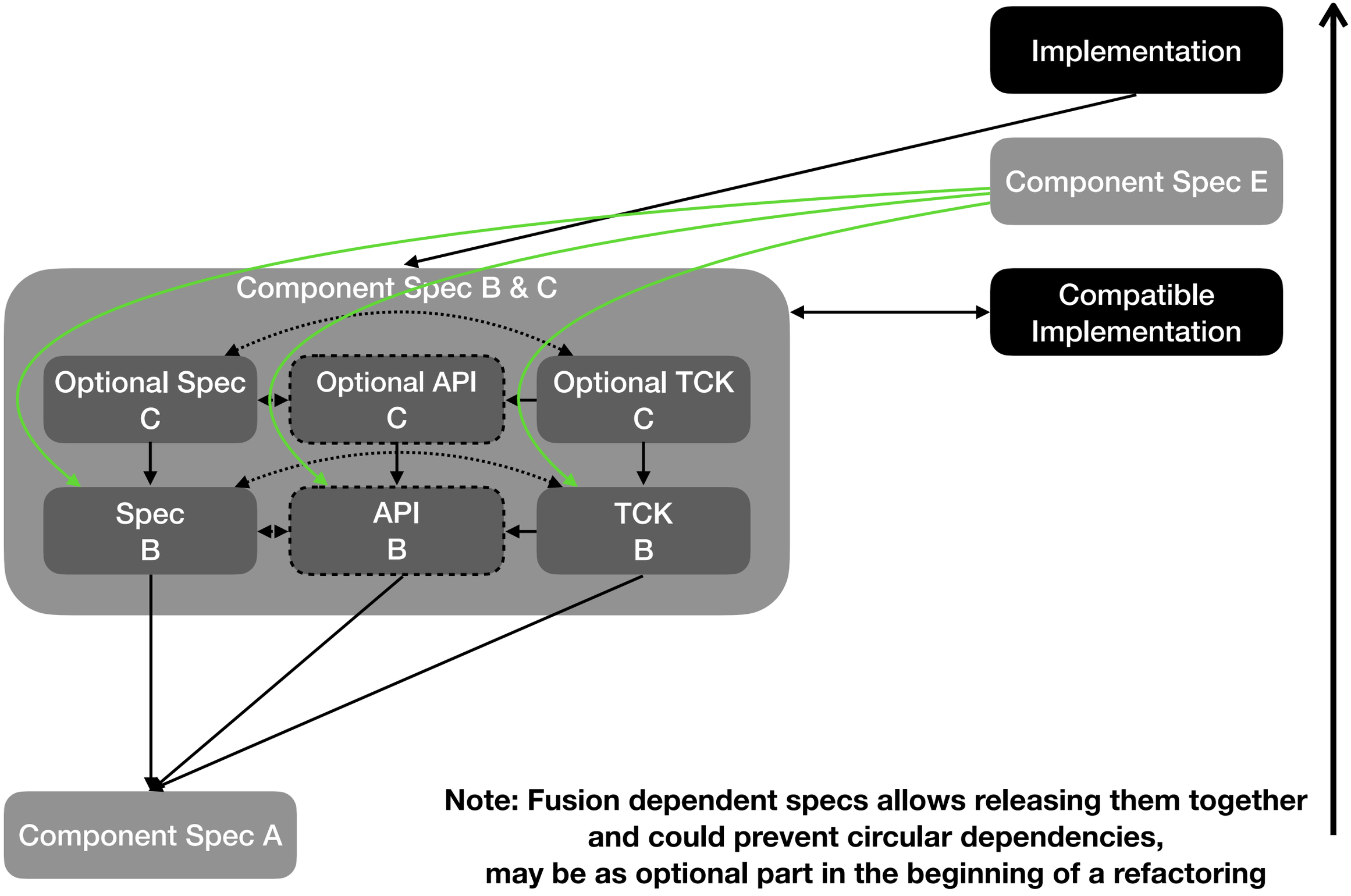
Note: Fusion dependent specs allows releasing them together and could prevent circular dependencies

Component Spec Detail



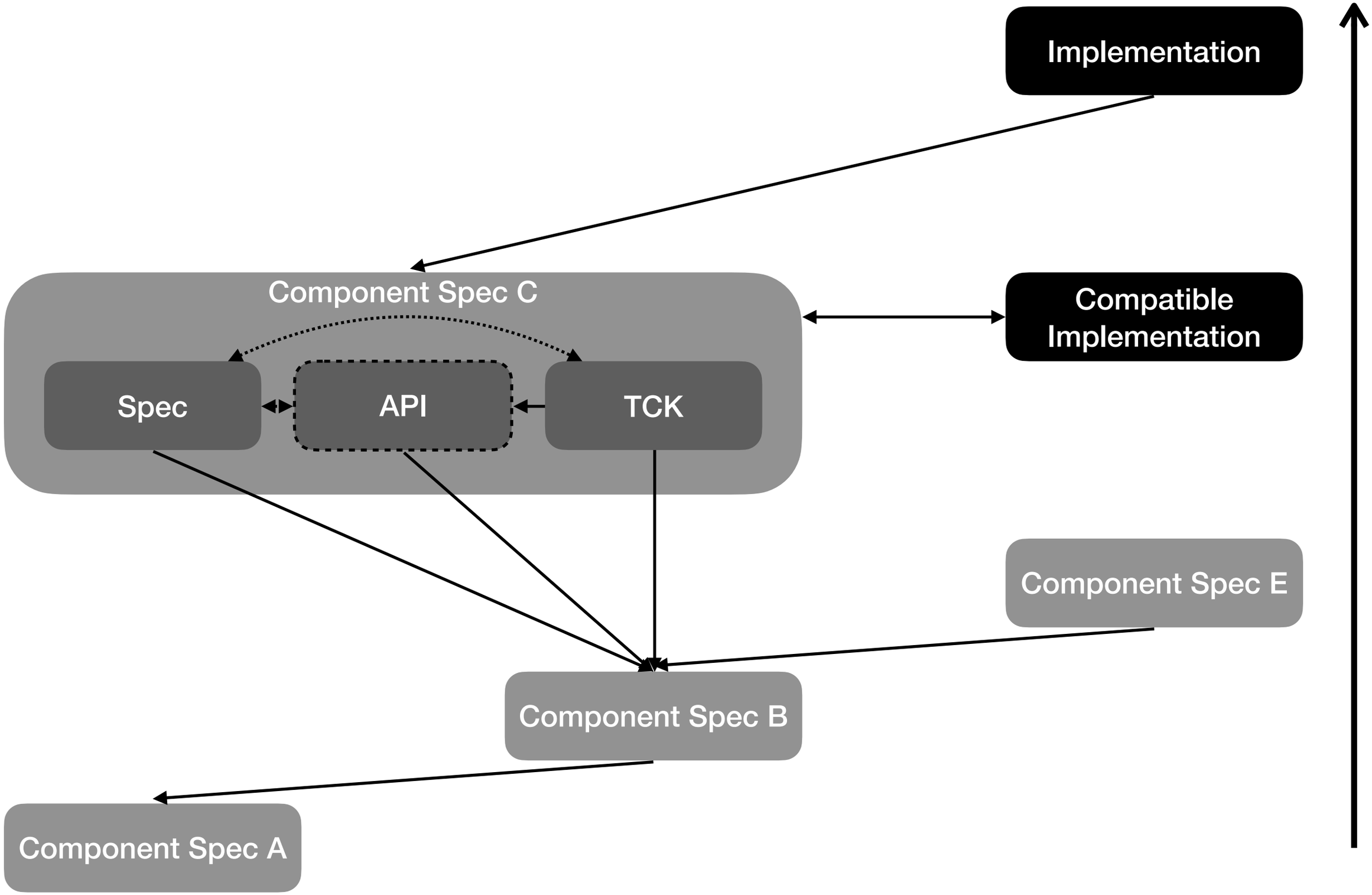
Note: Fusion dependent specs allows releasing them together and could prevent circular dependencies, may be as optional part in the beginning of a refactoring

Component Spec Detail

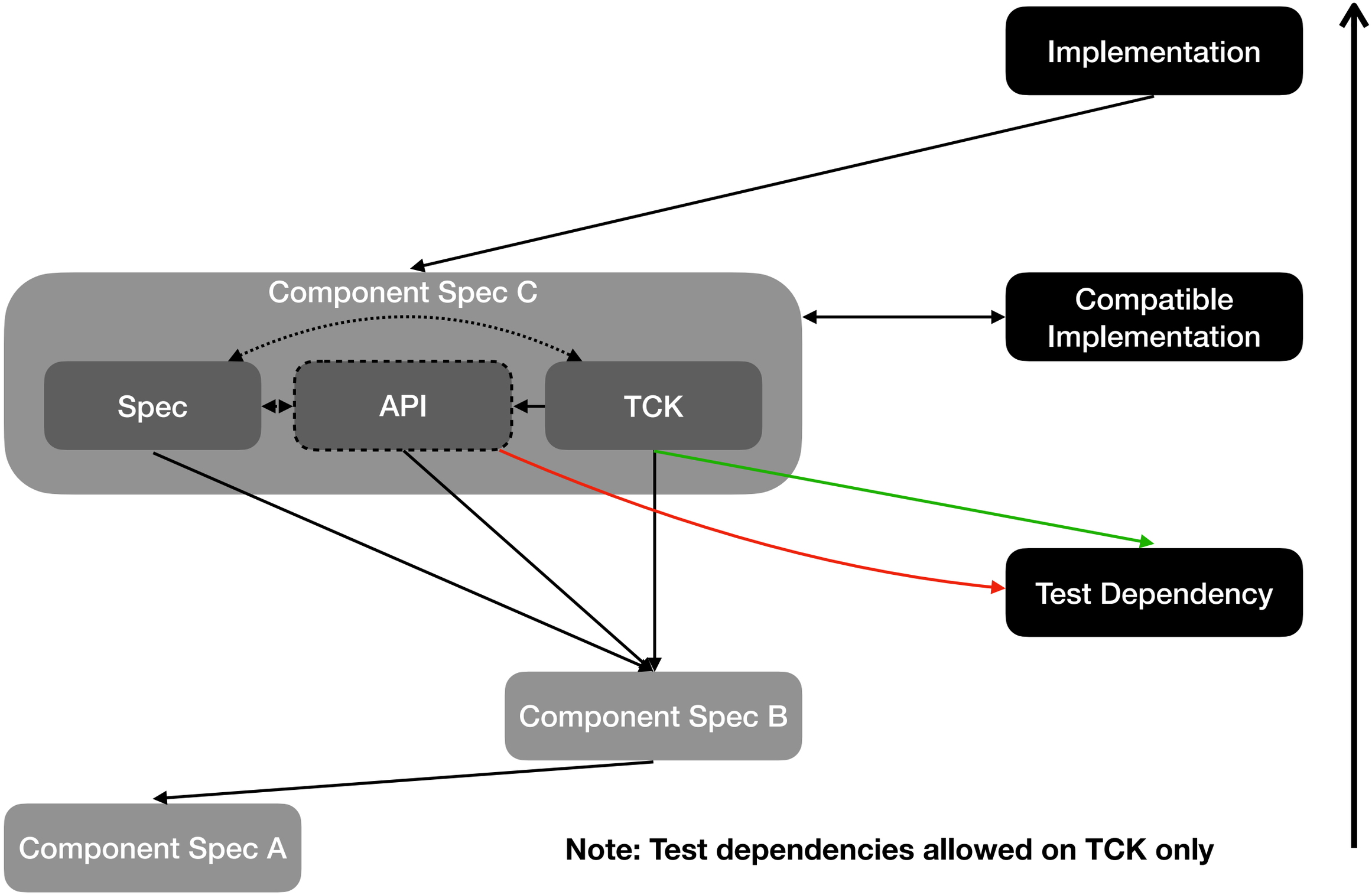


Note: Fusion dependent specs allows releasing them together and could prevent circular dependencies, may be as optional part in the beginning of a refactoring

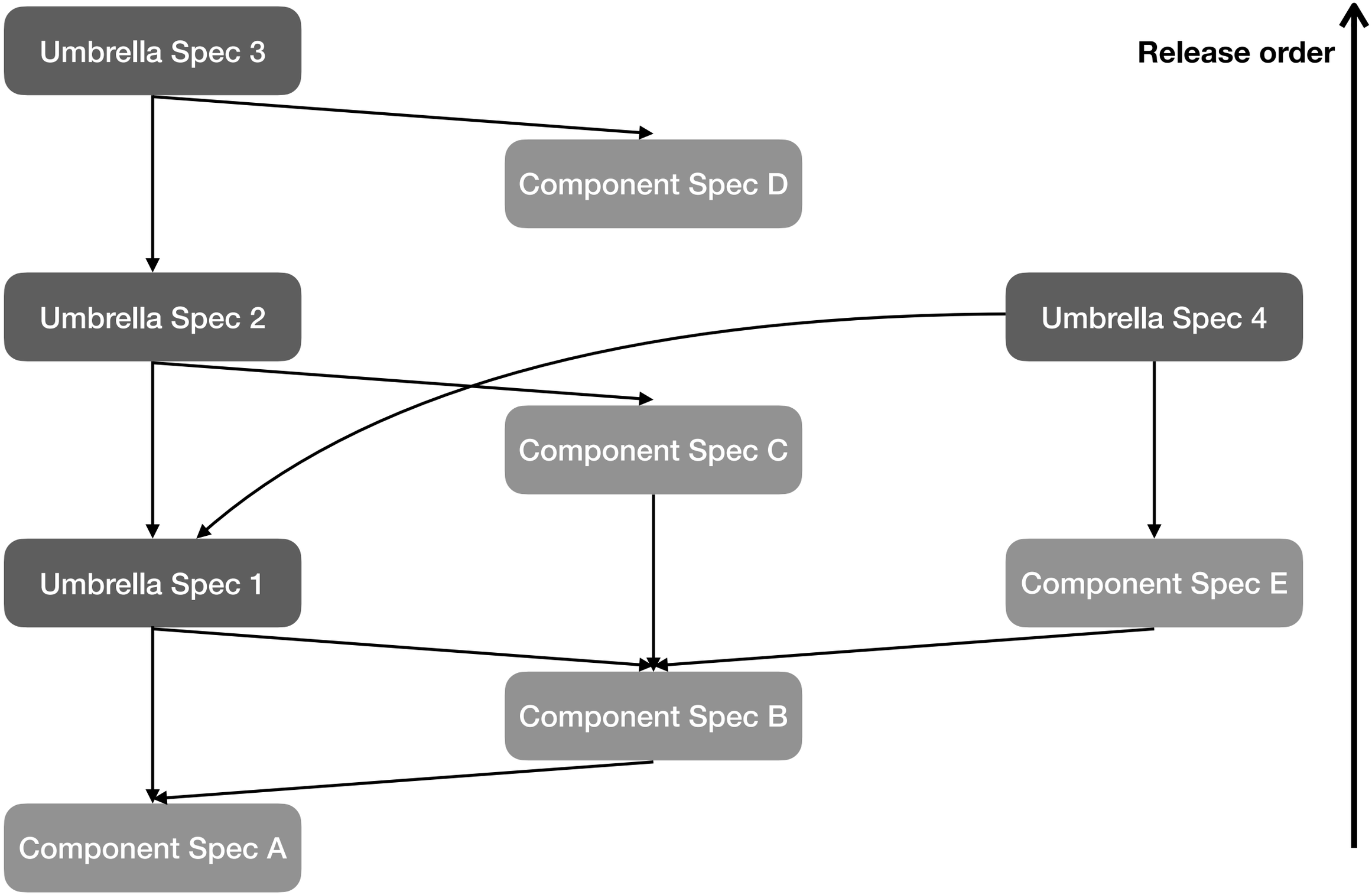
Component Spec Detail



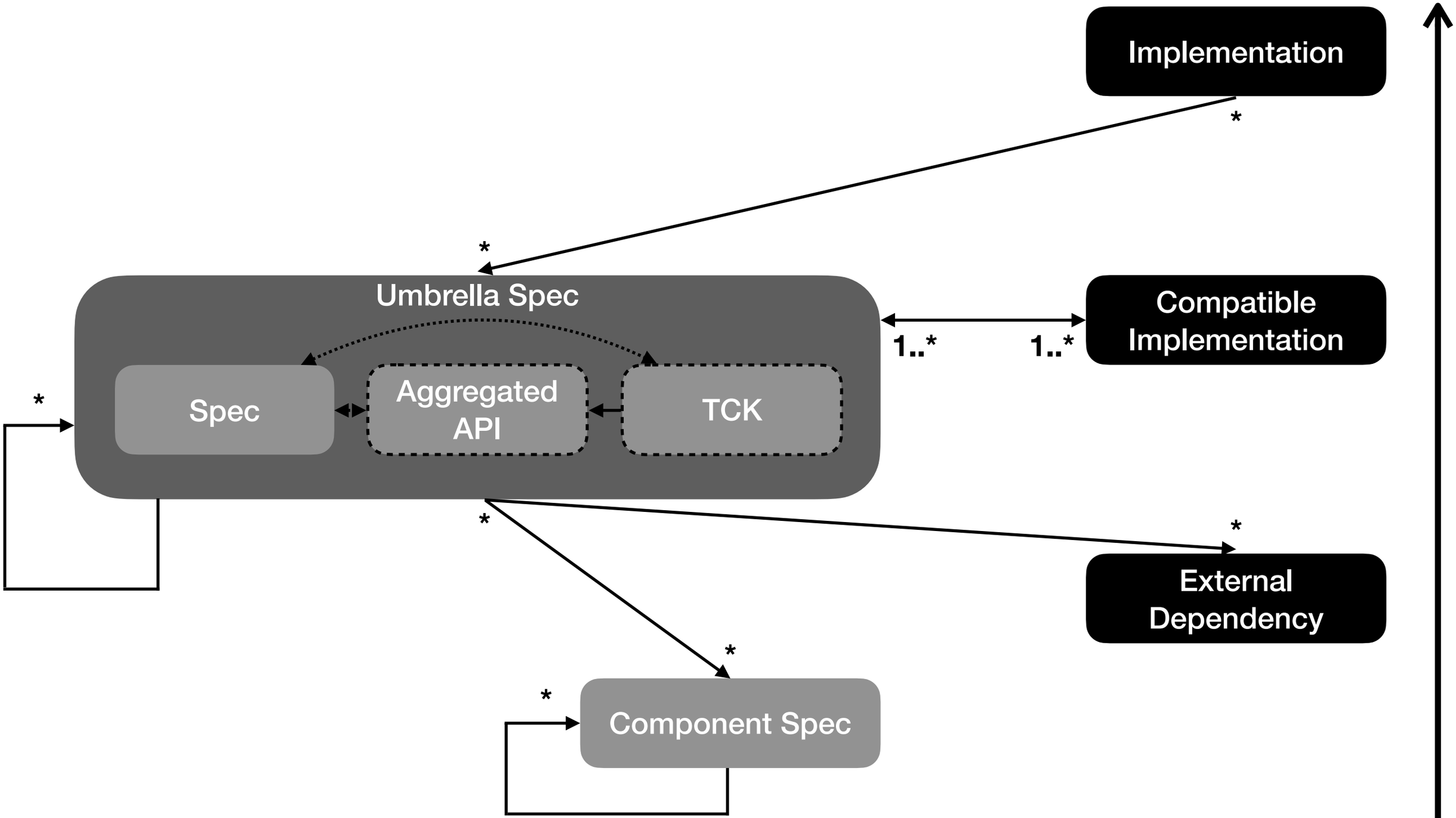
Component Spec Detail



Umbrella Specs

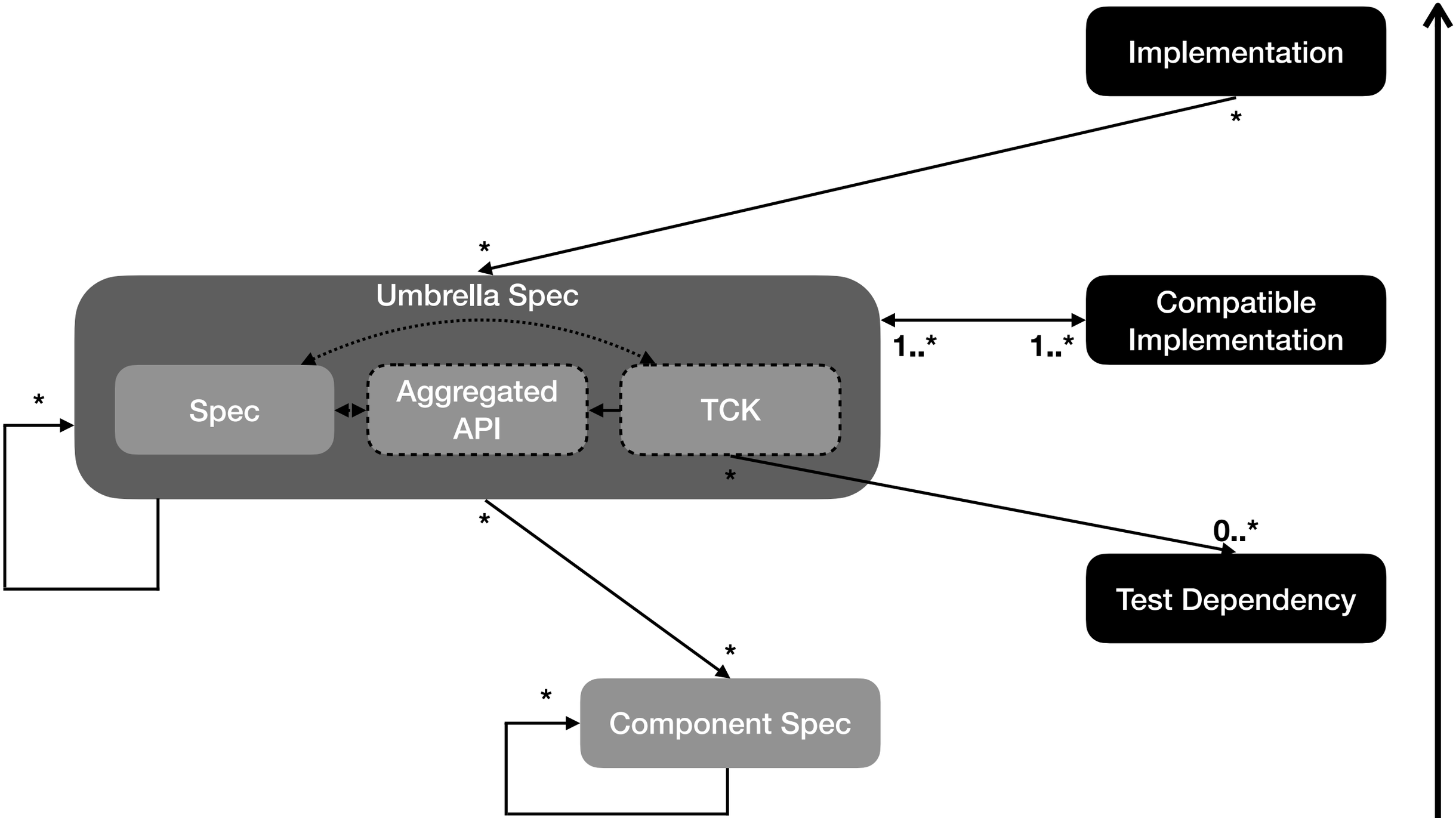


Umbrella Spec Detail



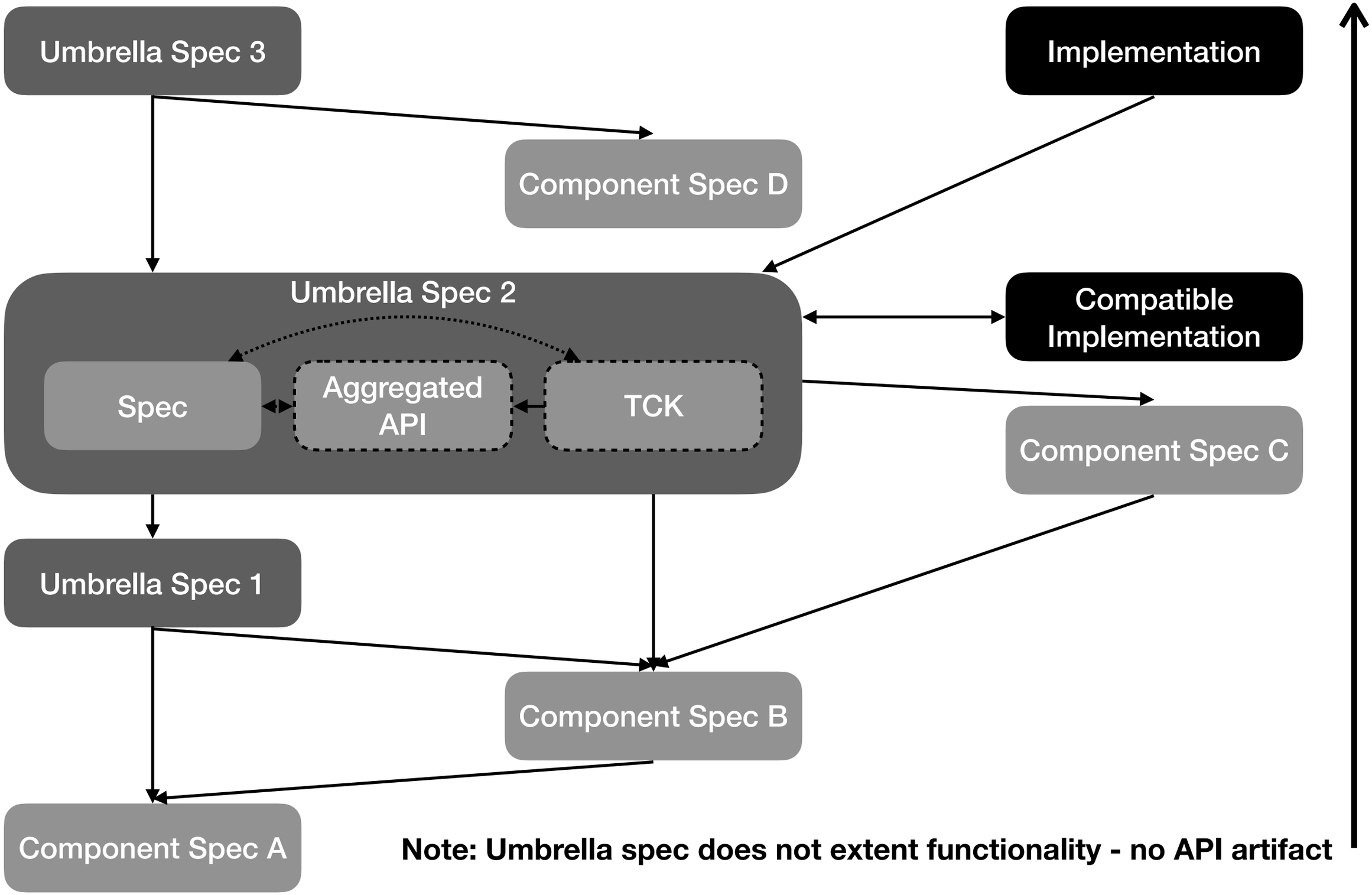
Note: Umbrella spec does not extent functionality - no source API artifact

Umbrella Spec Detail



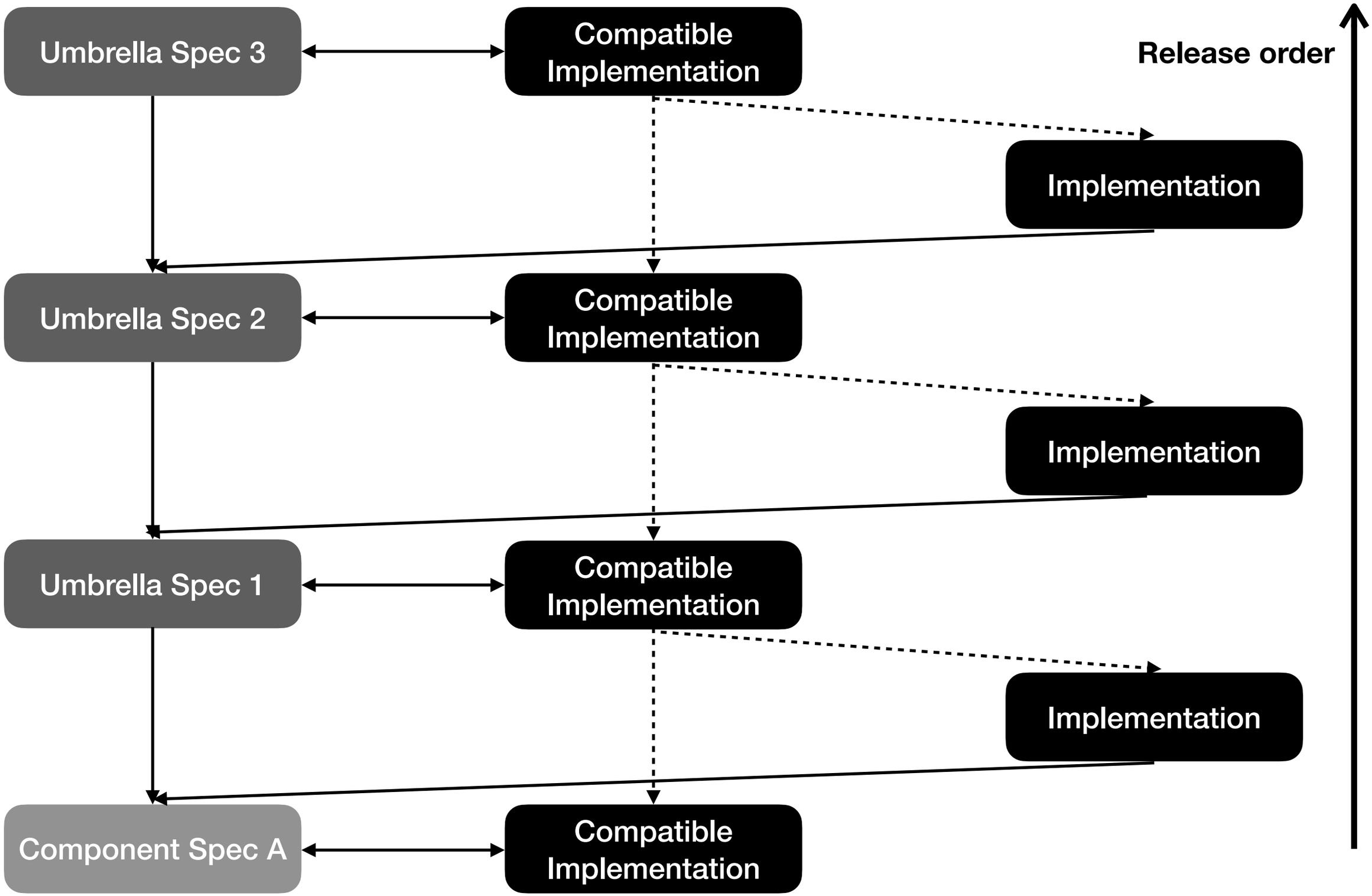
Note: Test dependencies allowed on TCK only

Umbrella Spec Detail

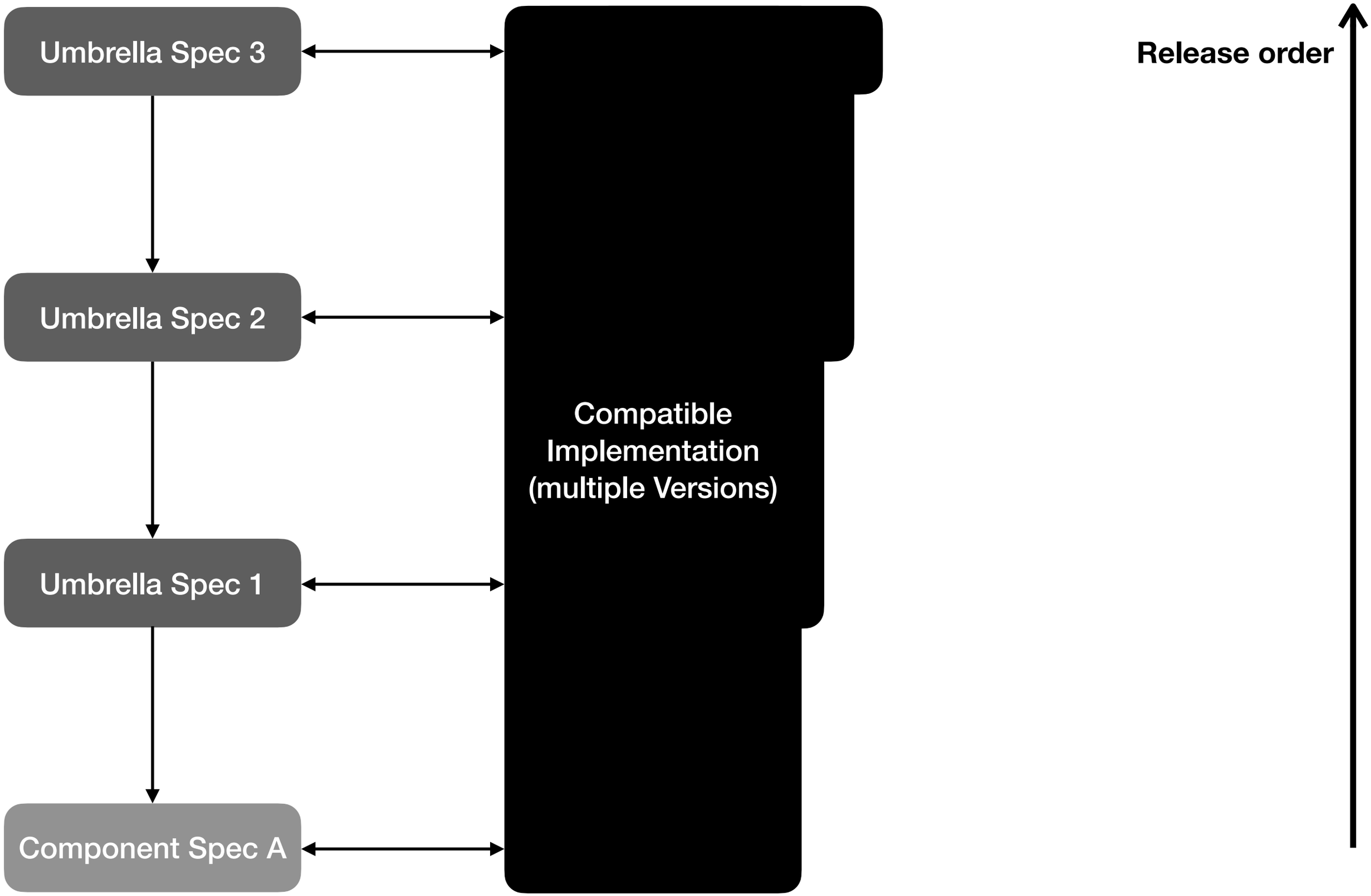


Note: Umbrella spec does not extent functionality - no API artifact

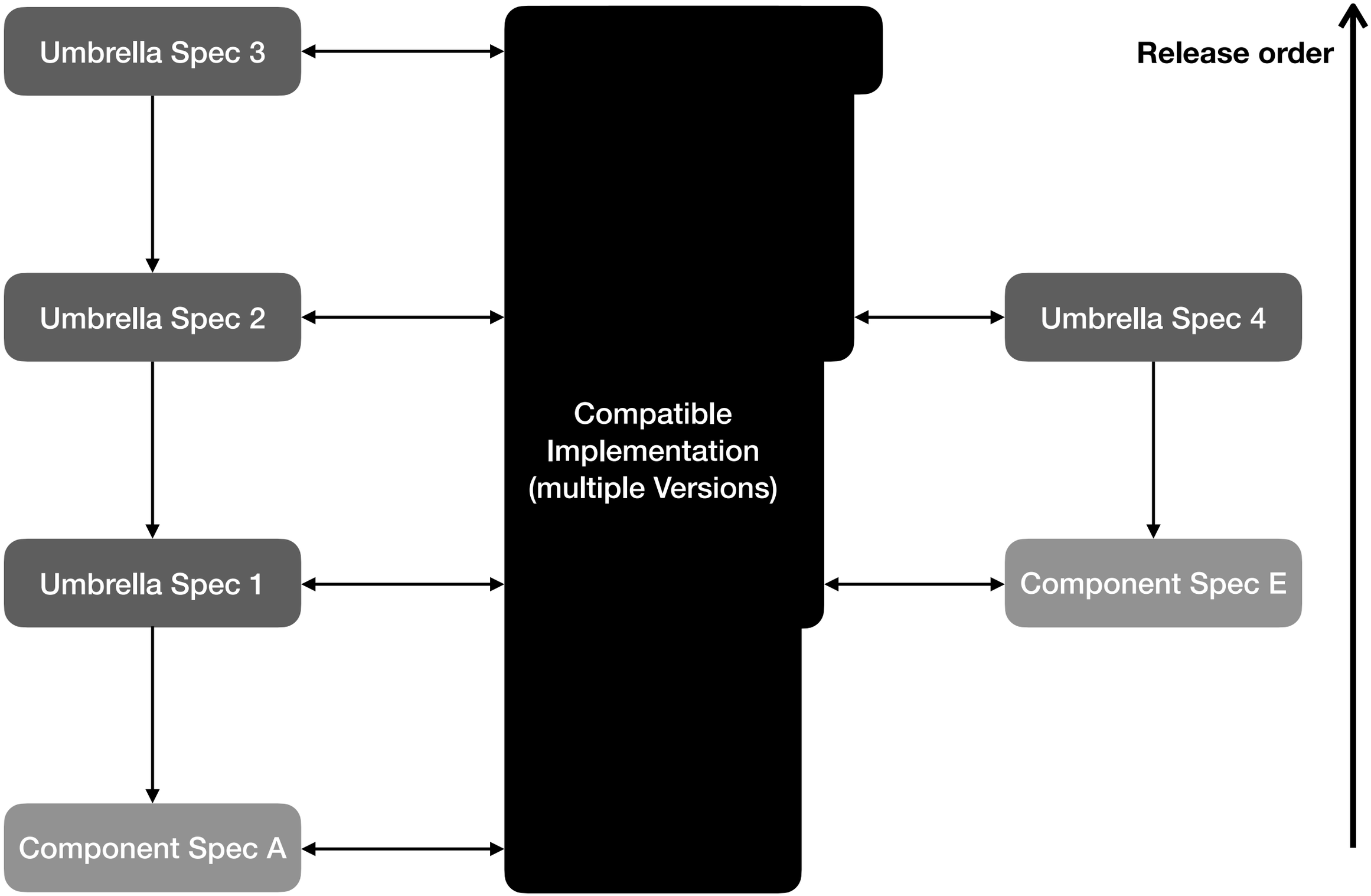
Spec Implementations



Spec Implementations



Spec Implementations

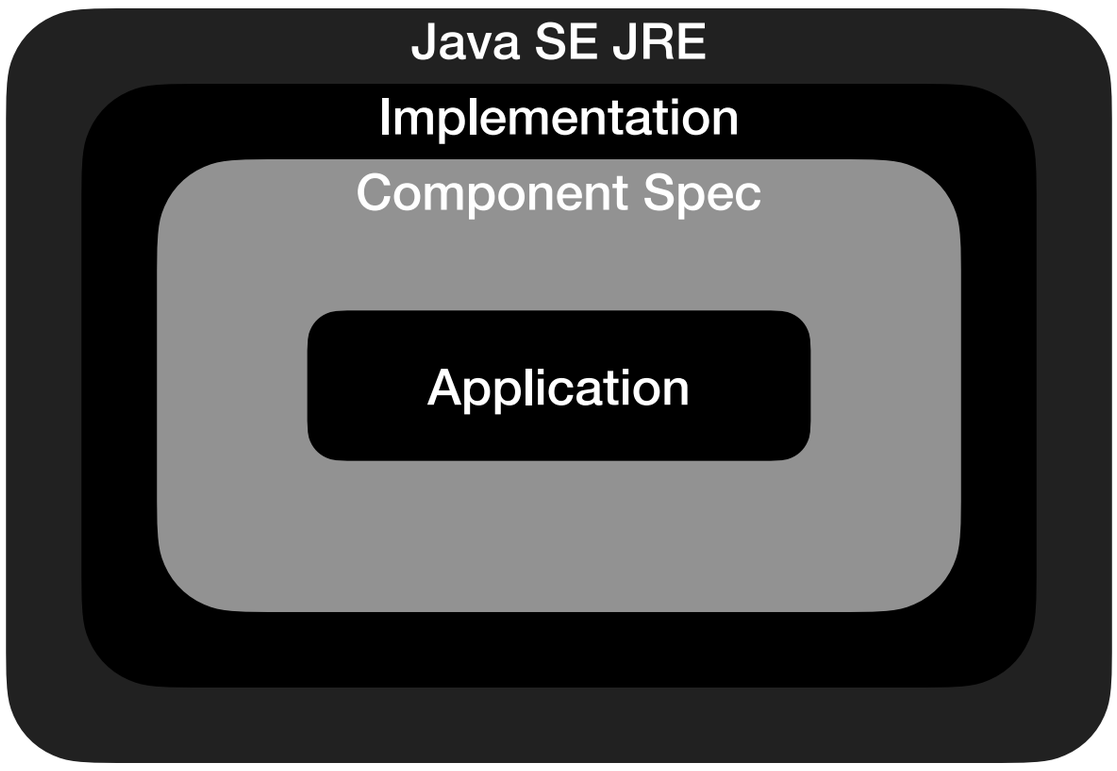


System Environment

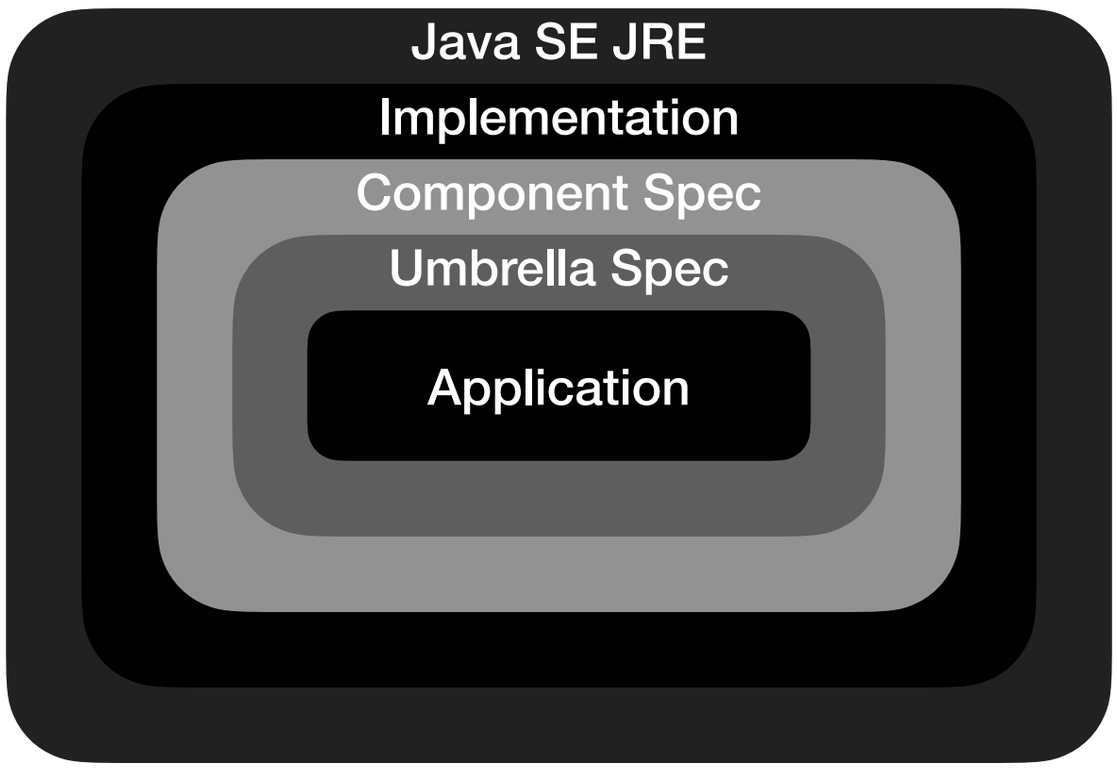
Application

System Environment

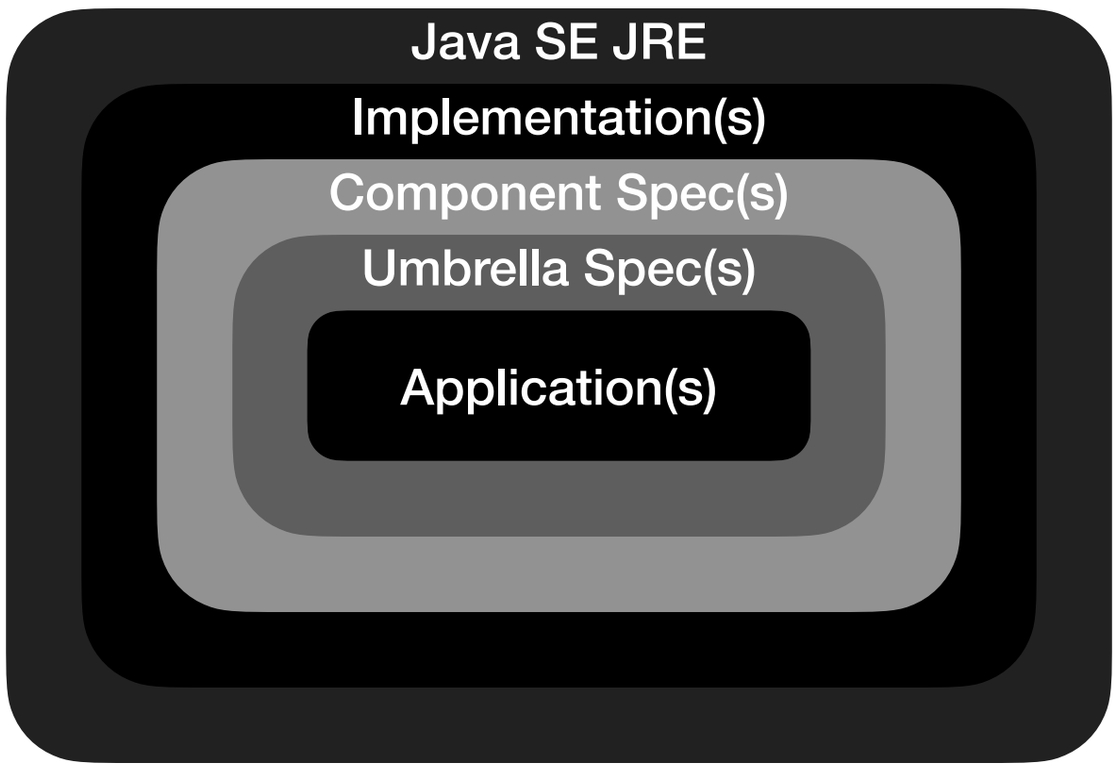
System Environment



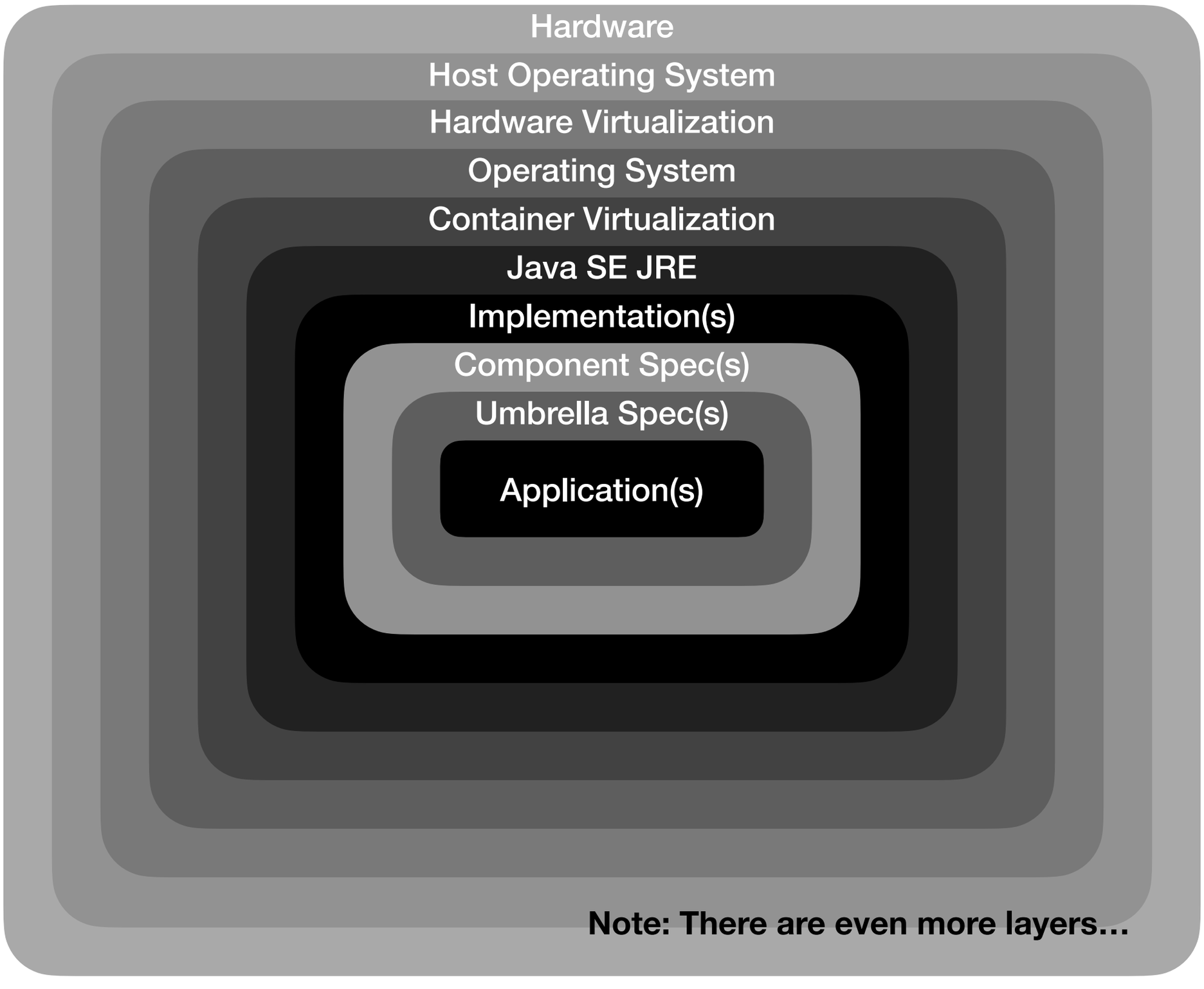
System Environment



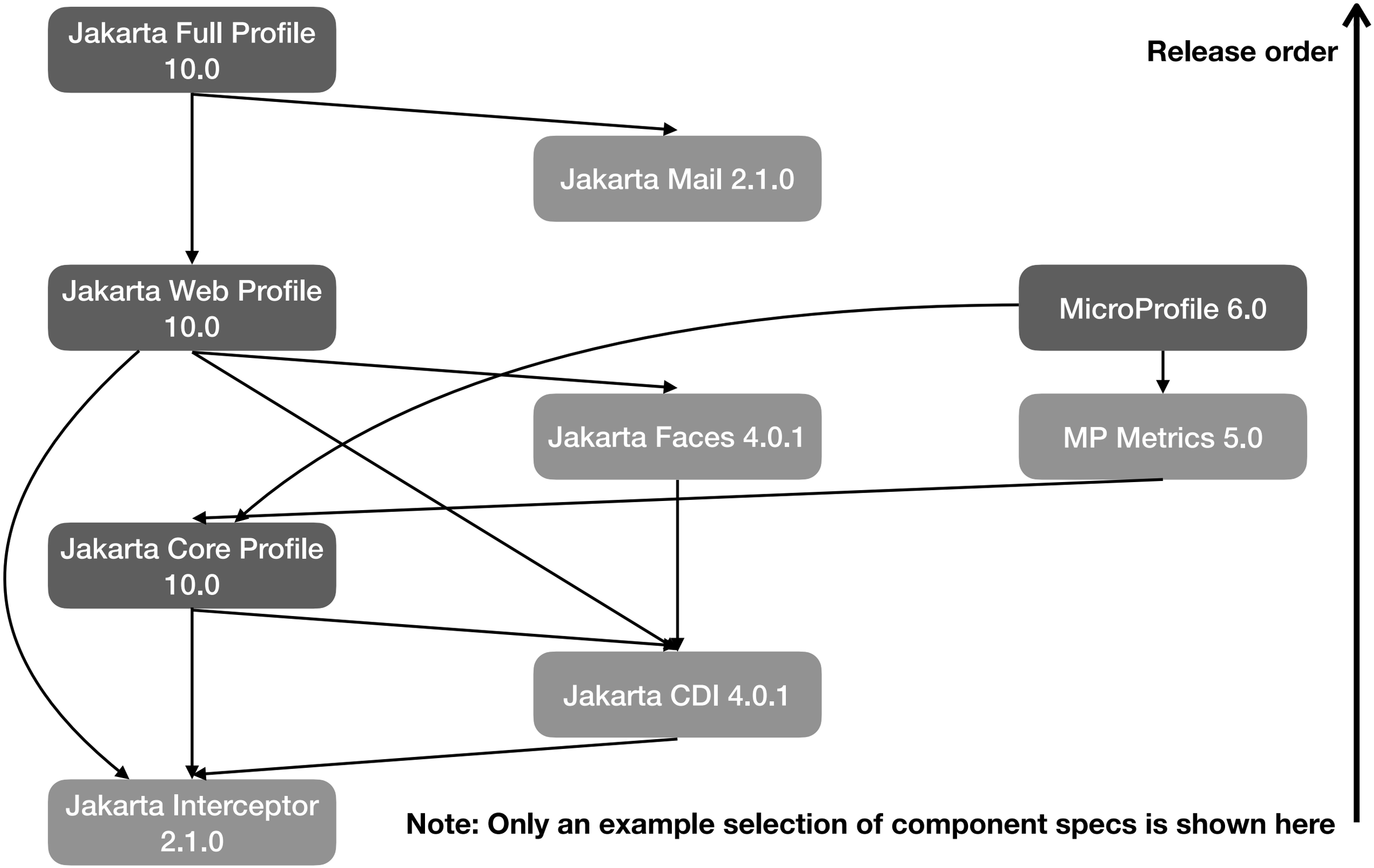
System Environment



System Environment

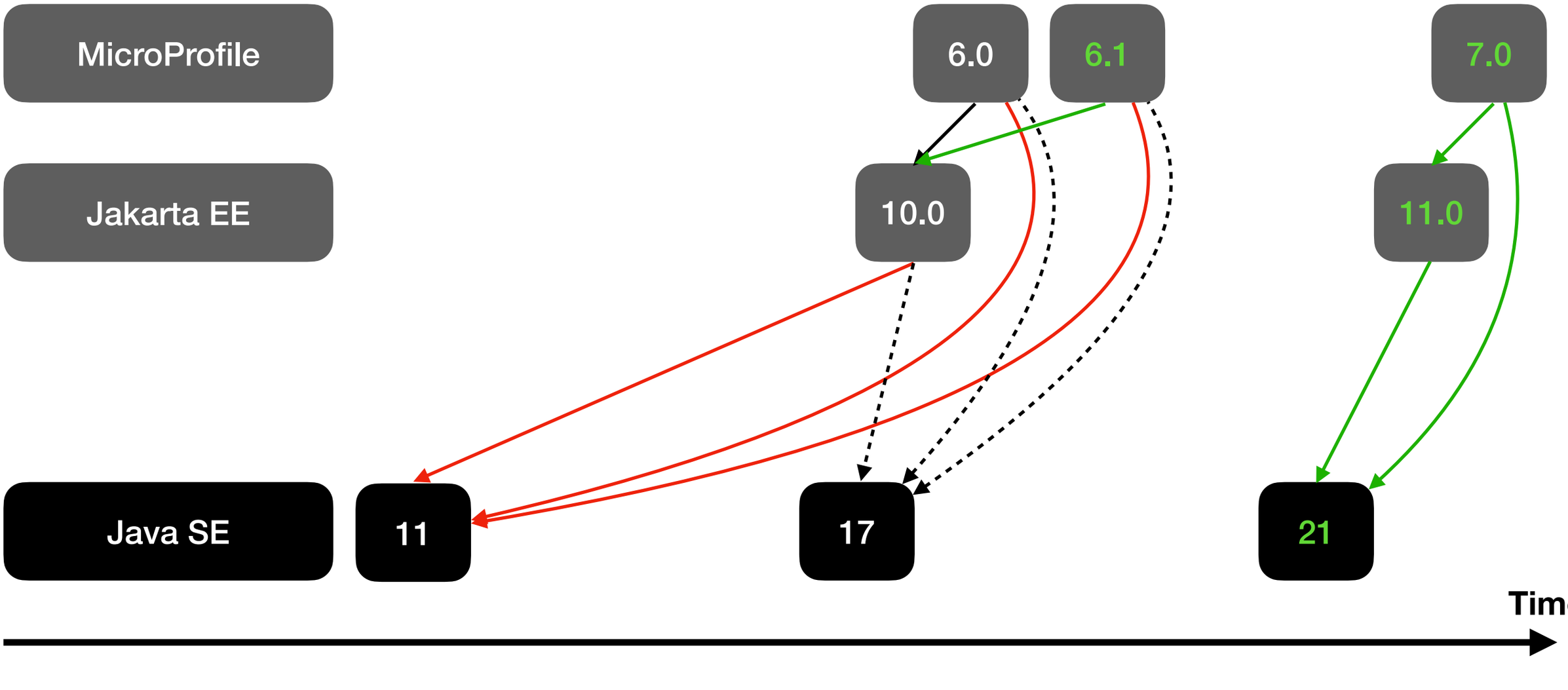


Jakarta EE & MicroProfile

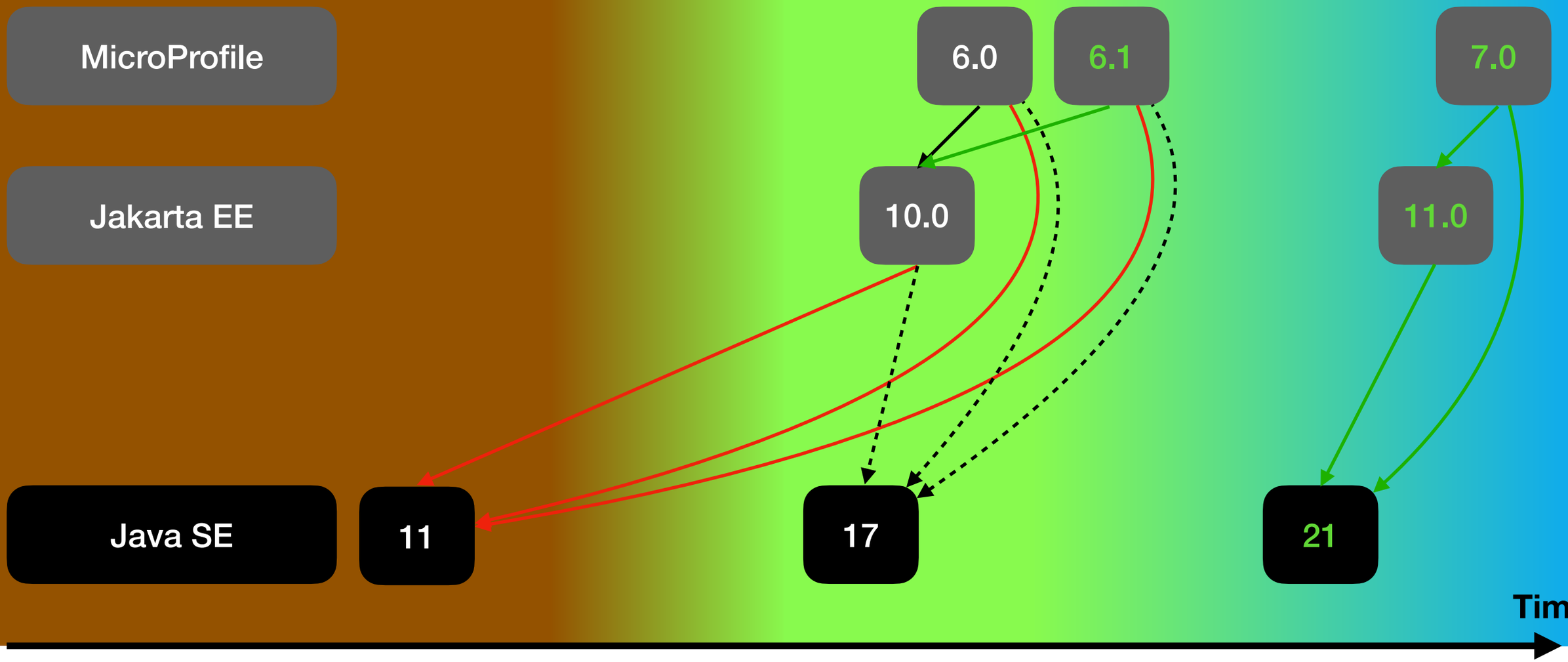


Note: Only an example selection of component specs is shown here

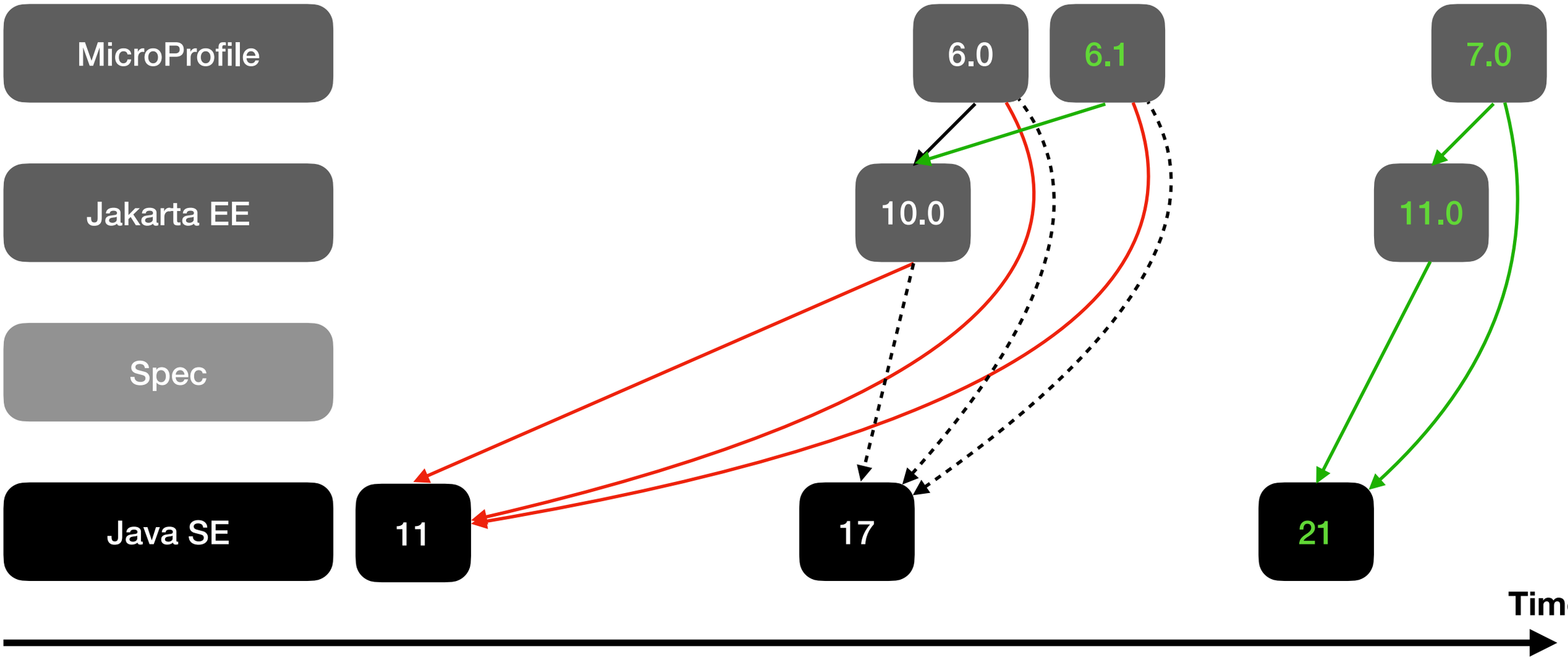
CN4J Roadmap



CN4J Roadmap



CN4J Roadmap



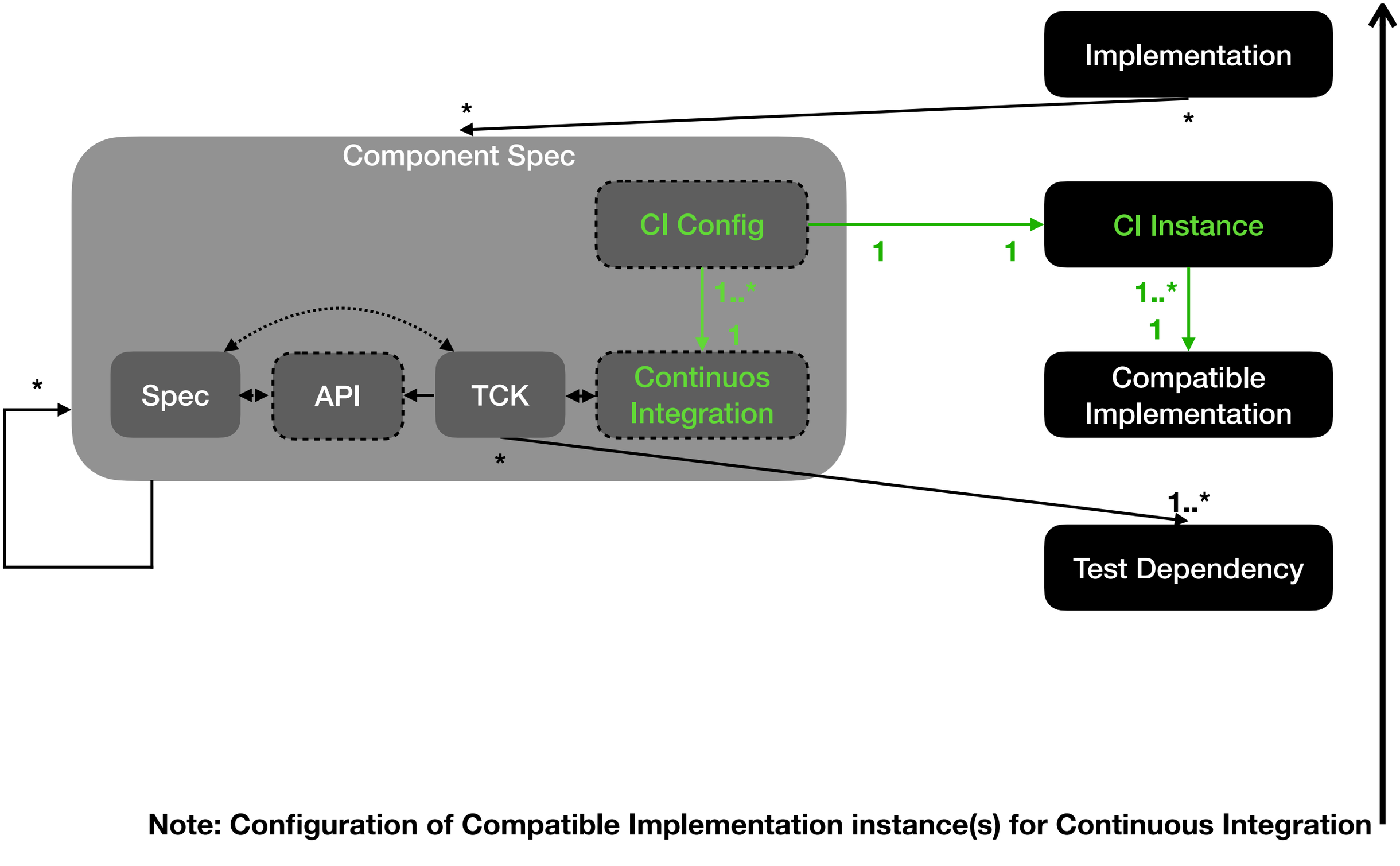
Issues & Solutions

- Continuous Integration
- MicroProfile Parent
- MicroProfile Spec
- Jakarta EE Parent
- Jakarta EE Spec
- Jakarta EE Platform
- MicroProfile Metrics
- Jakarta Security & MP JWT
- MP Config & Jakarta Config
- MP Telemetry & OpenTelemetry
- MP Metrics & MP Telemetry
- MP REST Client & Jakarta REST

Continuous Integration

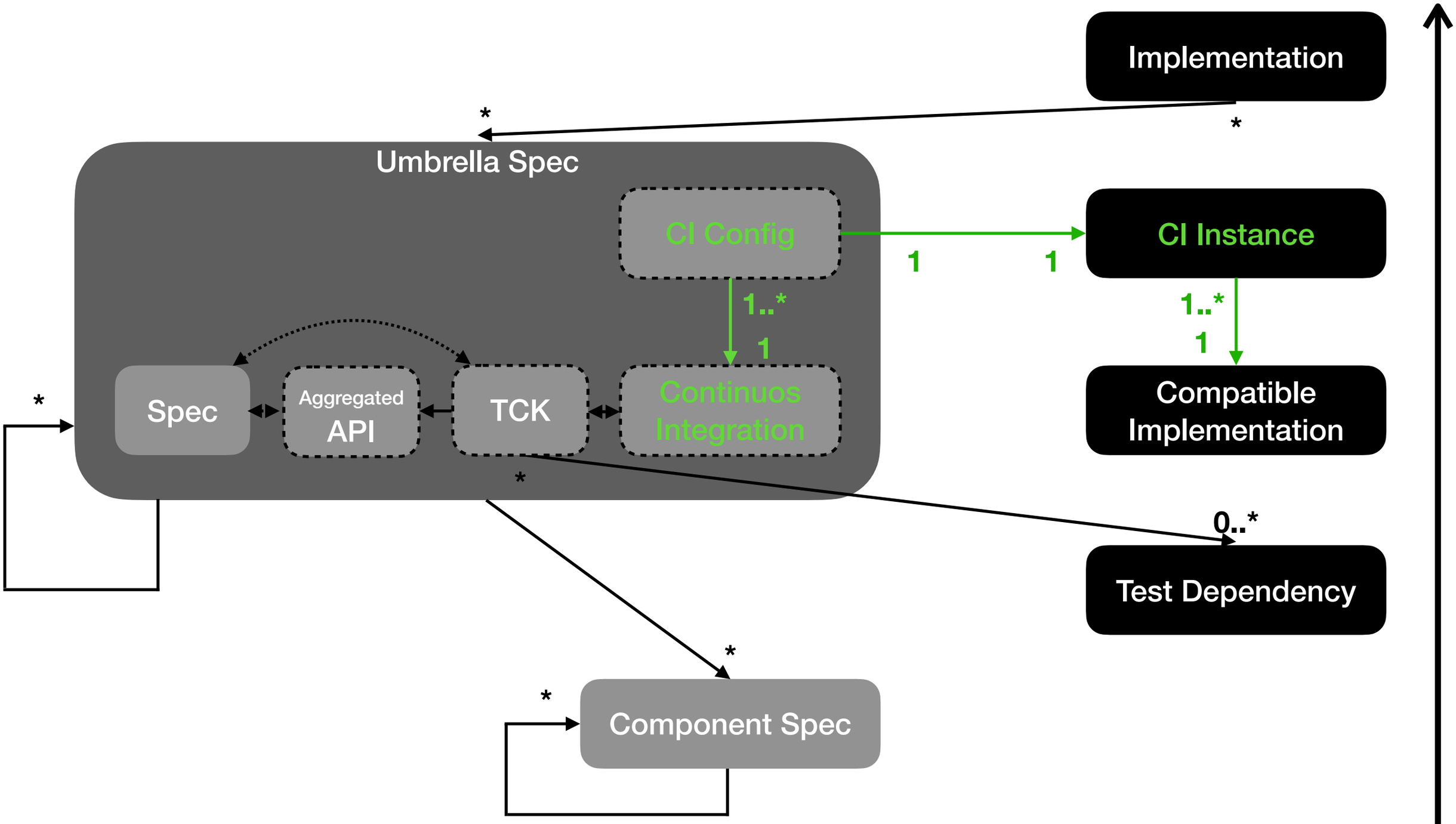
- Run TCK on Compatible Implementation(s) automatically and locally
  - Git clone & build CI or (Maven) load CI
  - Start CI & deploy TCK
  - Execute TCK in or with CI
  - Return TCK feedback

Continuous Integration



Note: Configuration of Compatible Implementation instance(s) for Continuous Integration

Continuous Integration

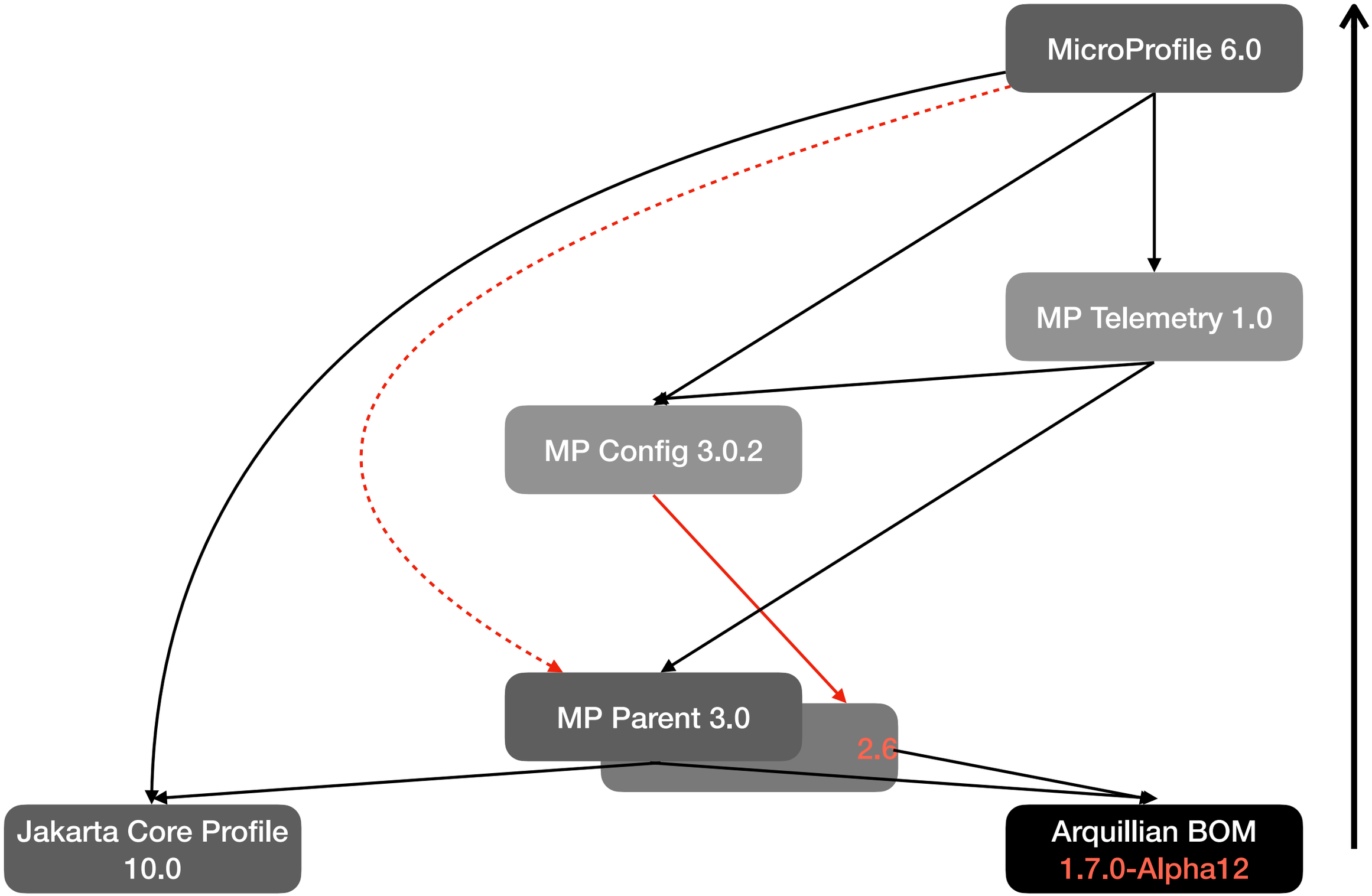


Note: Configuration of Compatible Implementation instance(s) for Continuous Integration

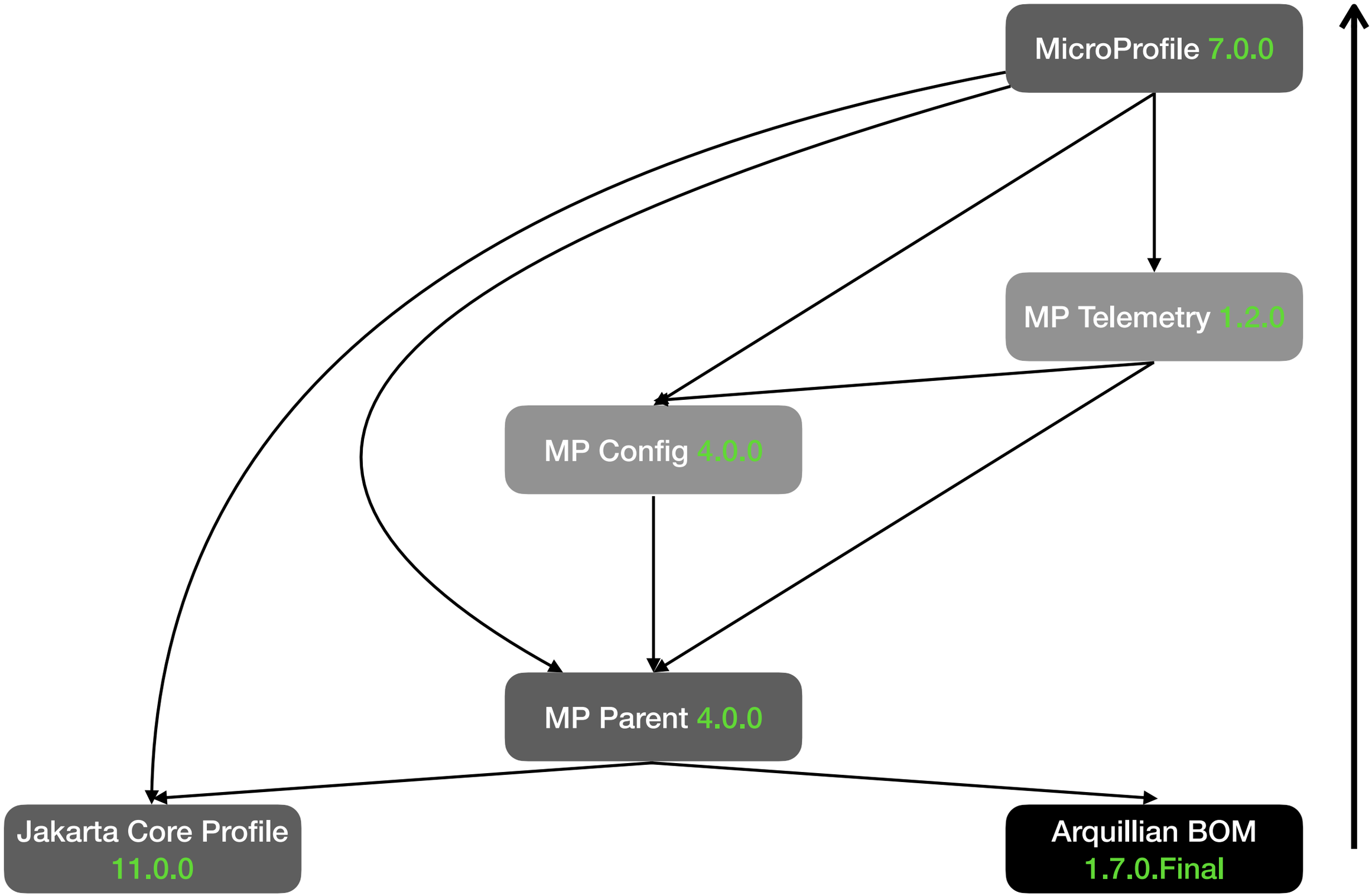
MicroProfile Parent

- Issue
  - Two versions in use for MP 6.0
  - MicroProfile (umbrella) spec does not use it
    - One MP component spec still uses test dependency on API
- Solution
  - Use one version MP Parent for MP component and umbrella specs
  - TCK-BOM prevents test dependencies on non-TCKs

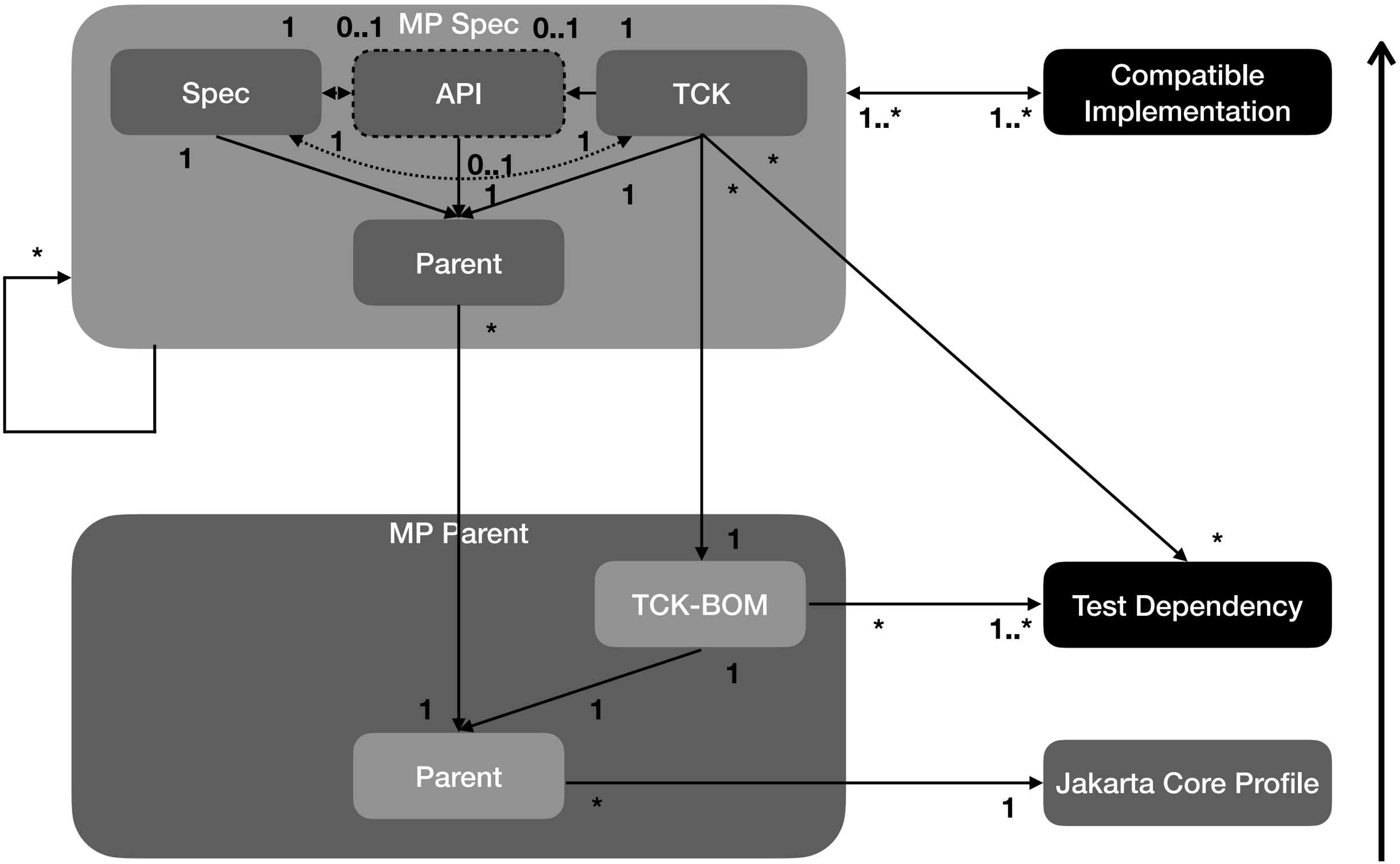
MicroProfile Parent



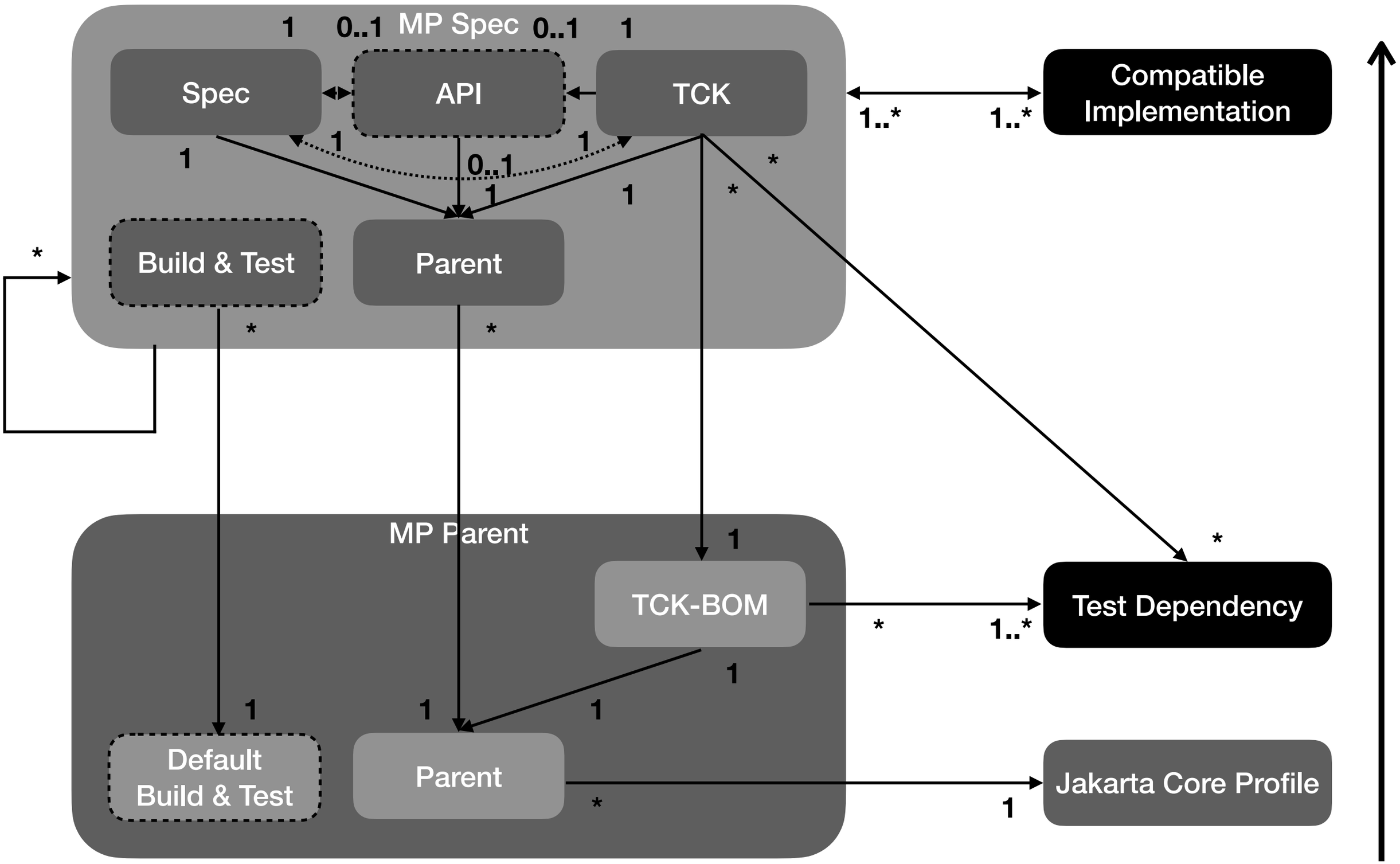
MicroProfile Parent



MicroProfile Parent Detail



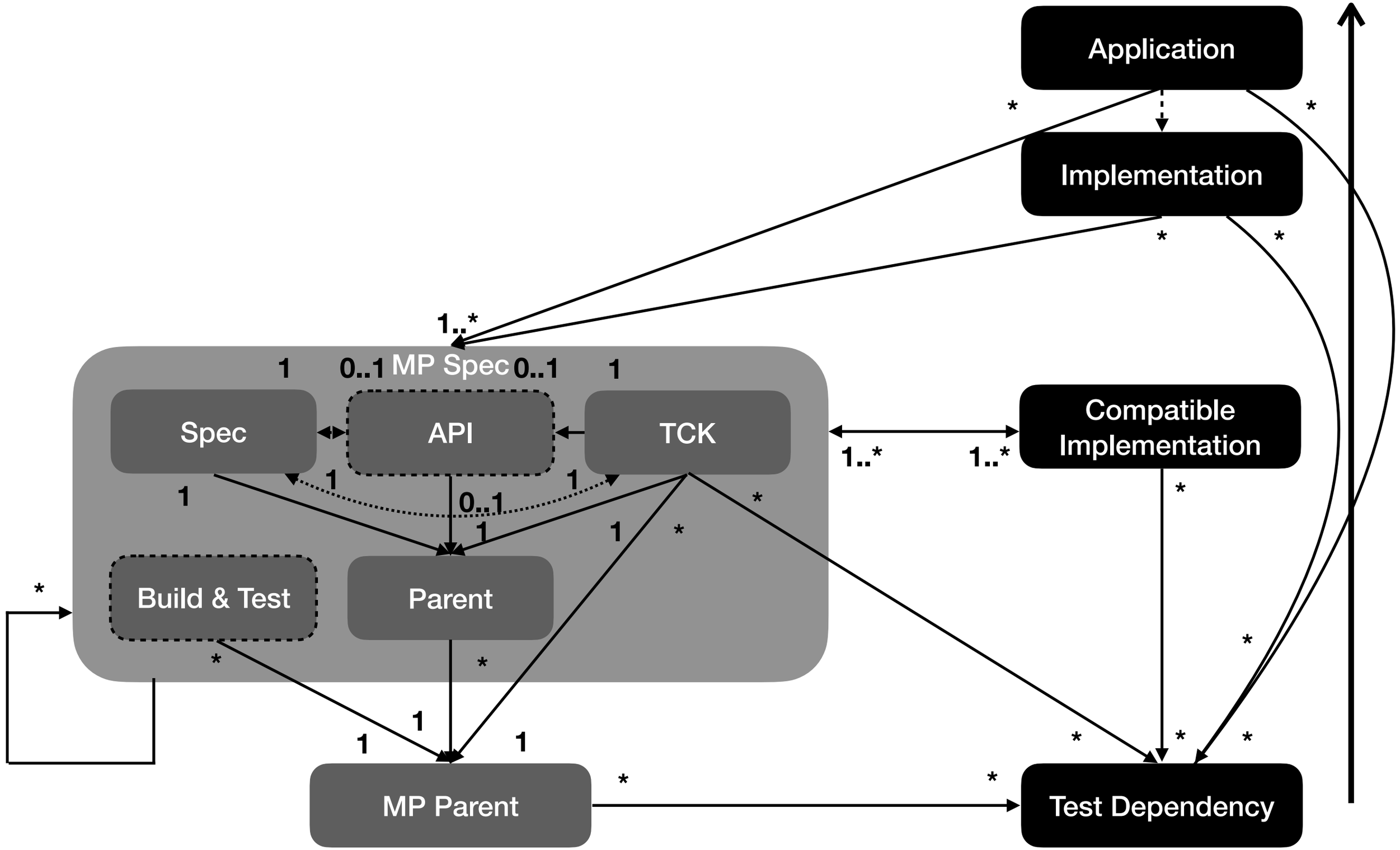
MicroProfile Parent Detail



MicroProfile Spec

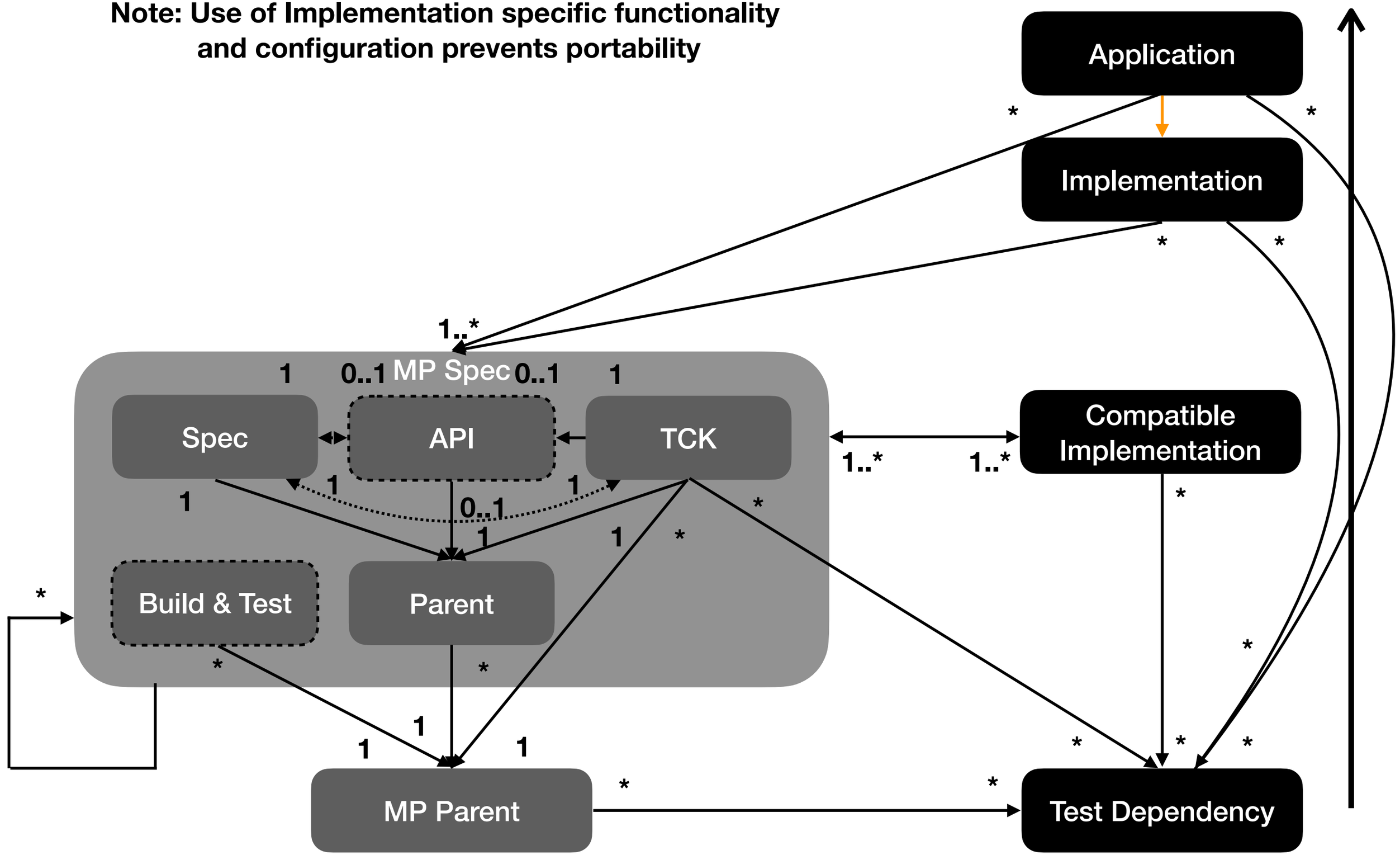
- Issue
    - Dependencies may deviate in component spec parts - testing deviating environment!
    - Dependencies may deviate in spec dependency graph - testing deviating environment!
        - Implementation tests may use component spec Namespace
- Solution
    - Use unified MP Parent version for all specs intended to be part one umbrella spec version
    - Keep all dependencies in sync for the spec dependency graph
    - Use unified naming for specs and components

MicroProfile Spec Detail



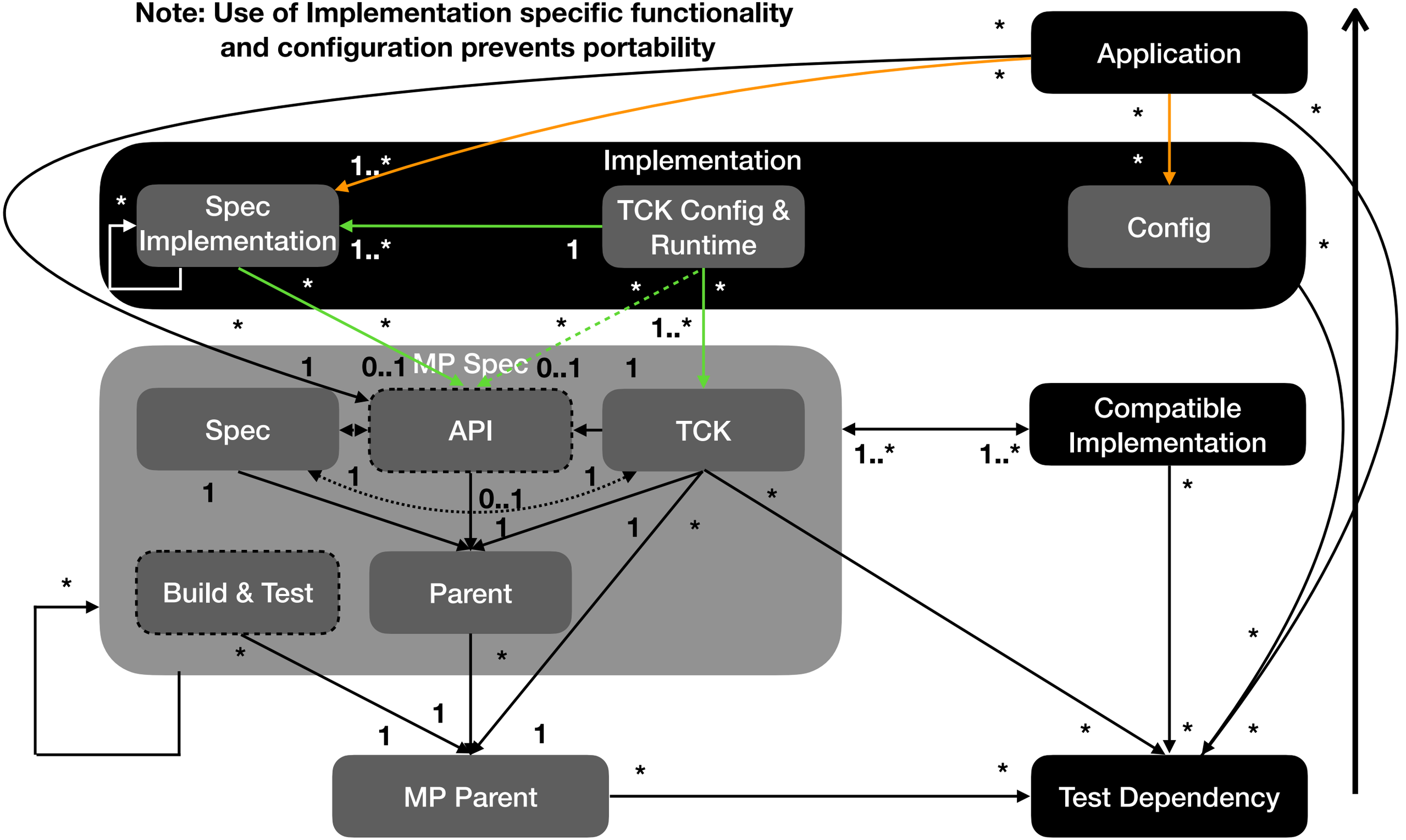
MicroProfile Spec Detail

Note: Use of Implementation specific functionality and configuration prevents portability



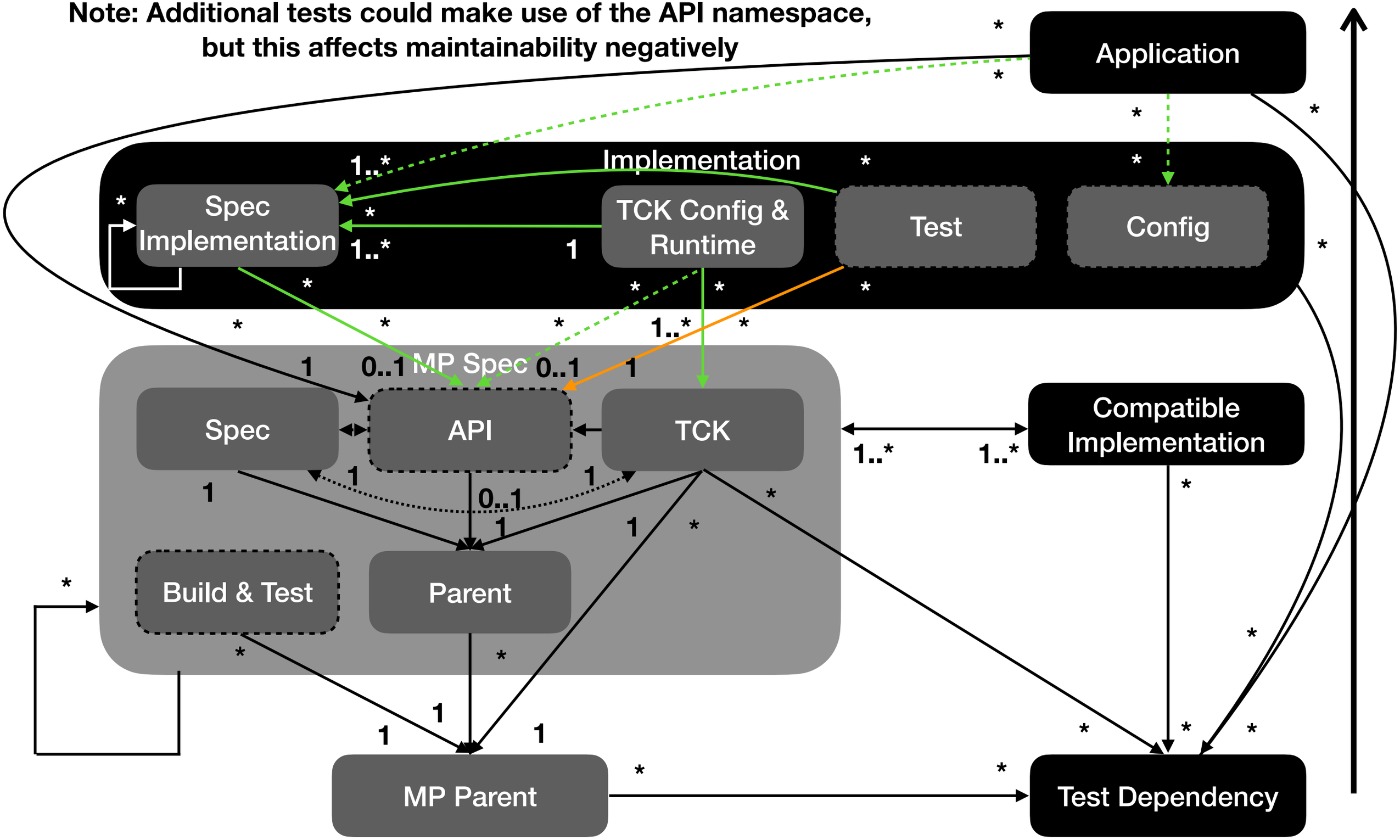
MicroProfile Spec Detail

Note: Use of Implementation specific functionality and configuration prevents portability



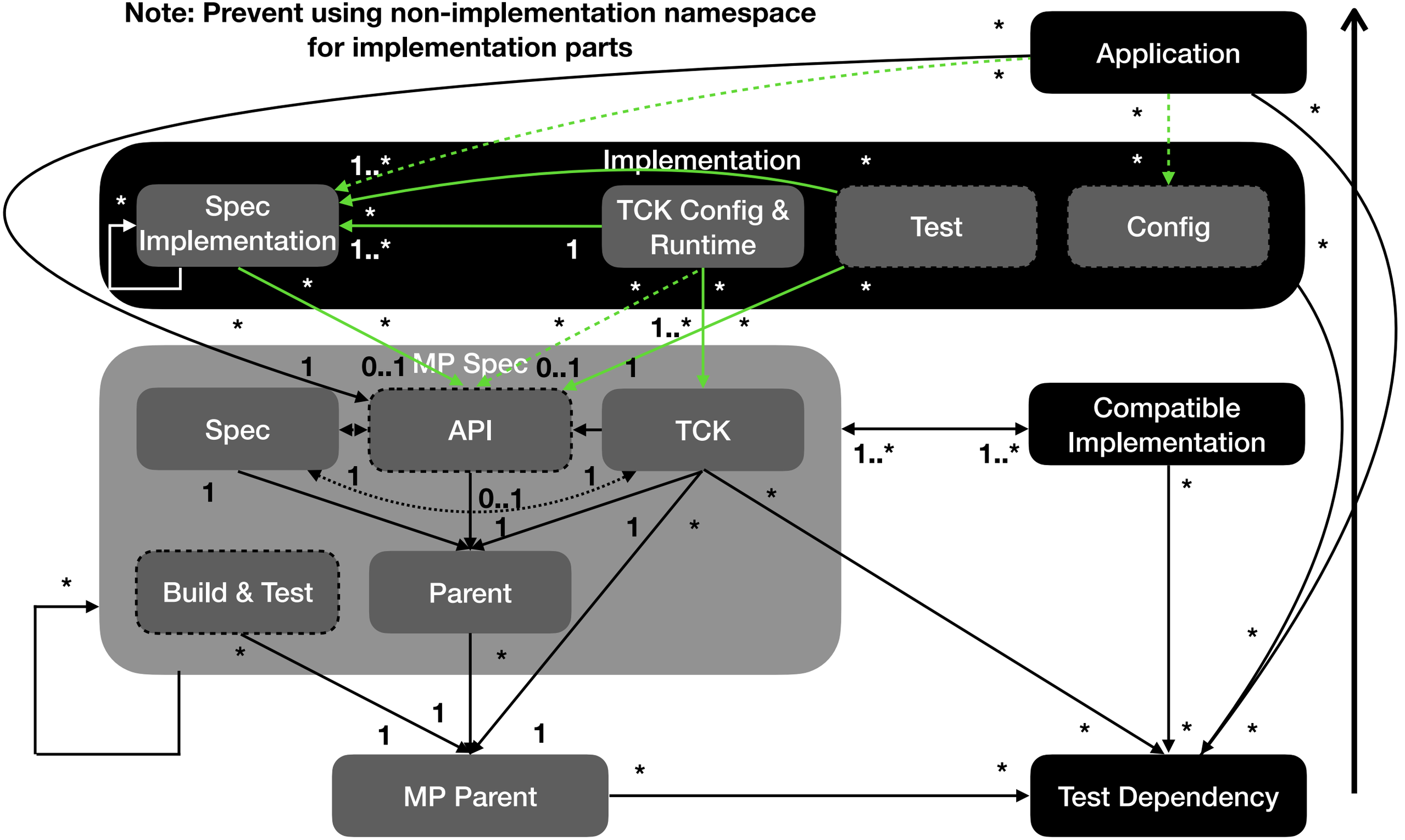
MicroProfile Spec Detail

Note: Additional tests could make use of the API namespace, but this affects maintainability negatively



MicroProfile Spec Detail

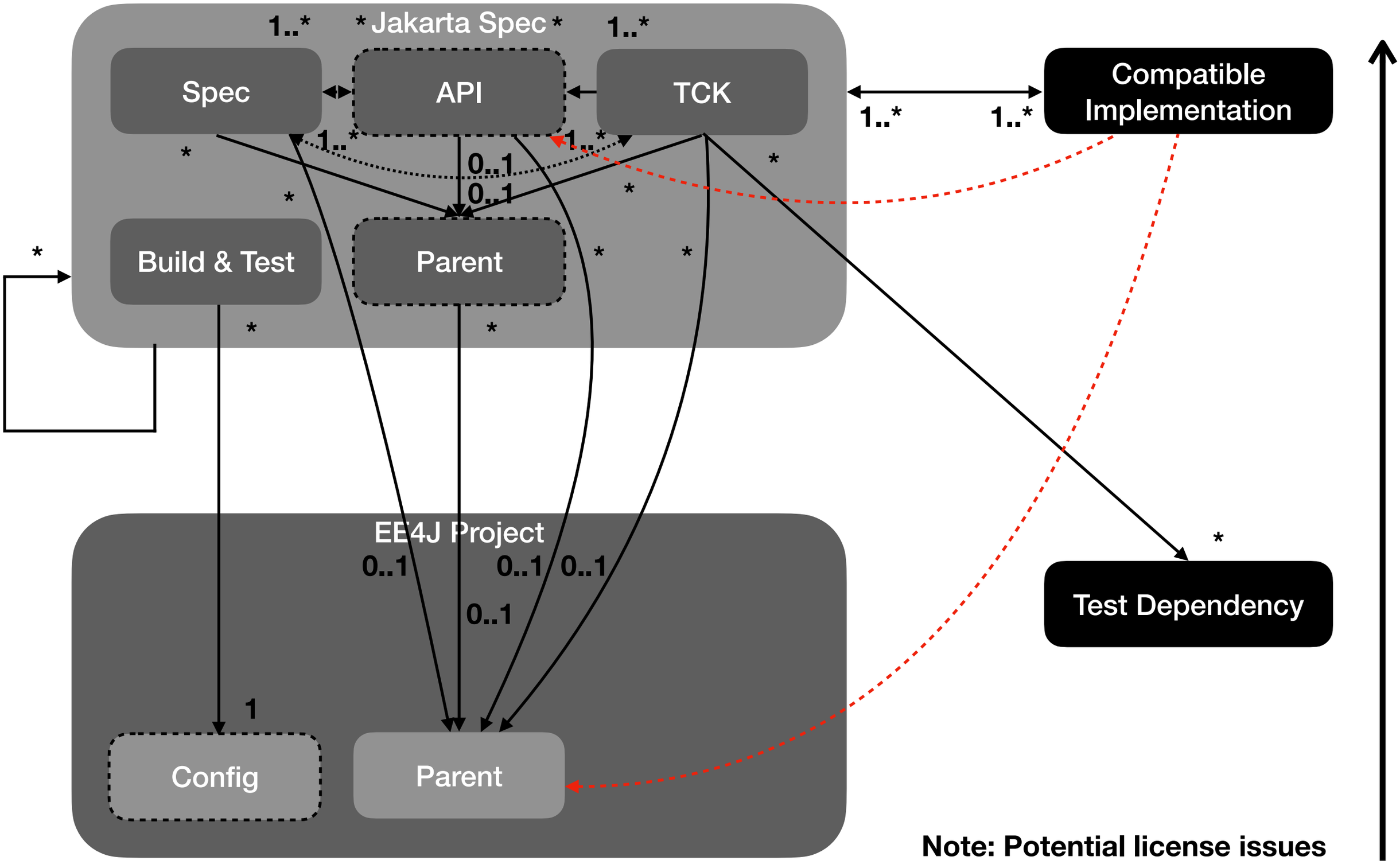
Note: Prevent using non-implementation namespace for implementation parts



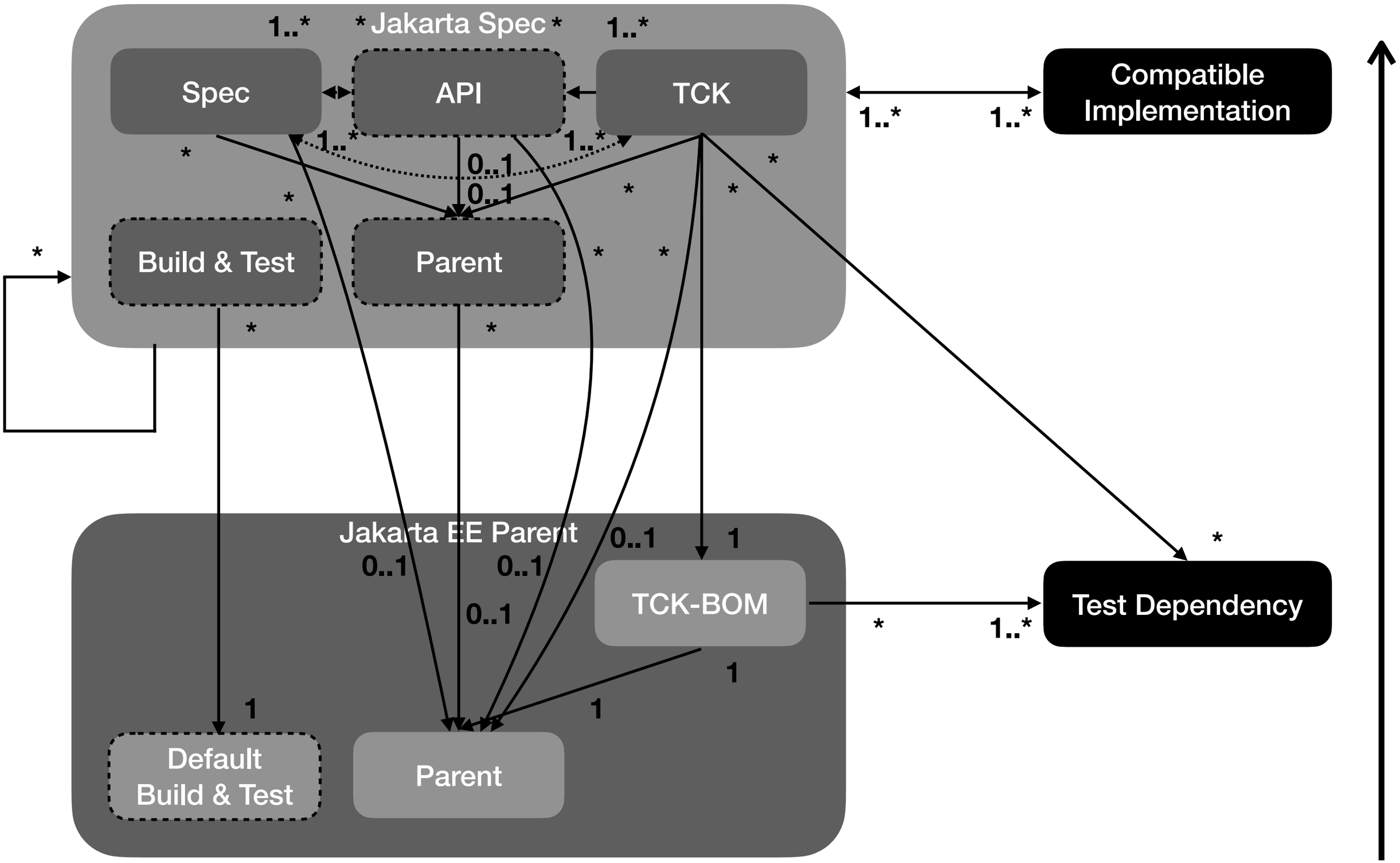
Jakarta EE Parent

- Issue
  - Defines configuration and licenses only
  - Test dependencies deviate in component specs
  - Build dependencies deviate in component specs
  - Deviating build configuration in specs
  - Name is EE4J Project, misleading when used as Jakarta (EE) Parent
- Solution
  - Add missing licenses to Jakarta EE Parent or create alternative Jakarta EE Parent (part)?
  - Add test dependencies to Jakarta EE Parent
  - Add build dependencies to Jakarta EE Parent
  - Add build configuration for specs
  - TCK-BOM prevents test dependencies on non-TCKs

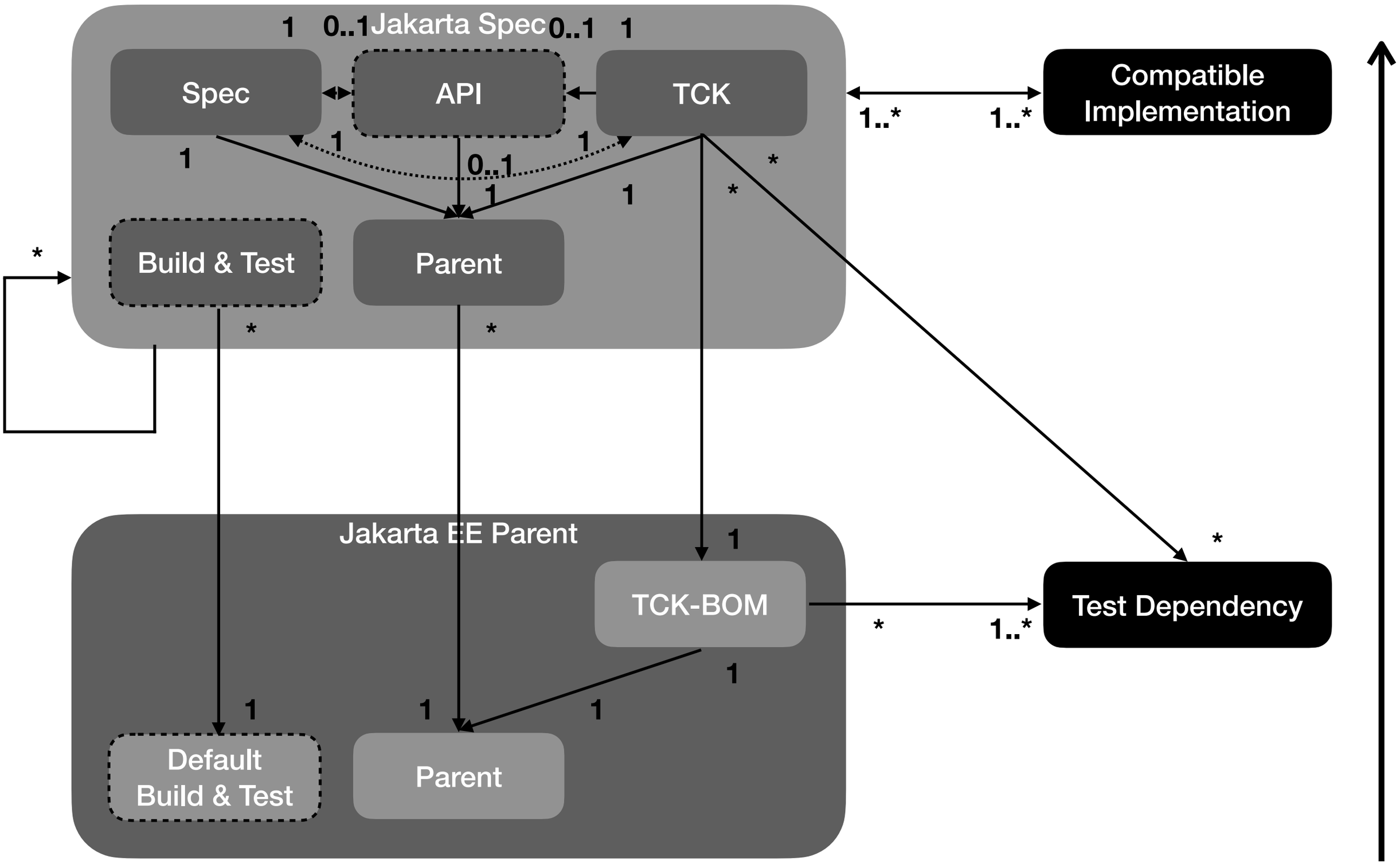
Jakarta EE Parent Detail



Jakarta EE Parent Detail



Jakarta EE Parent Detail

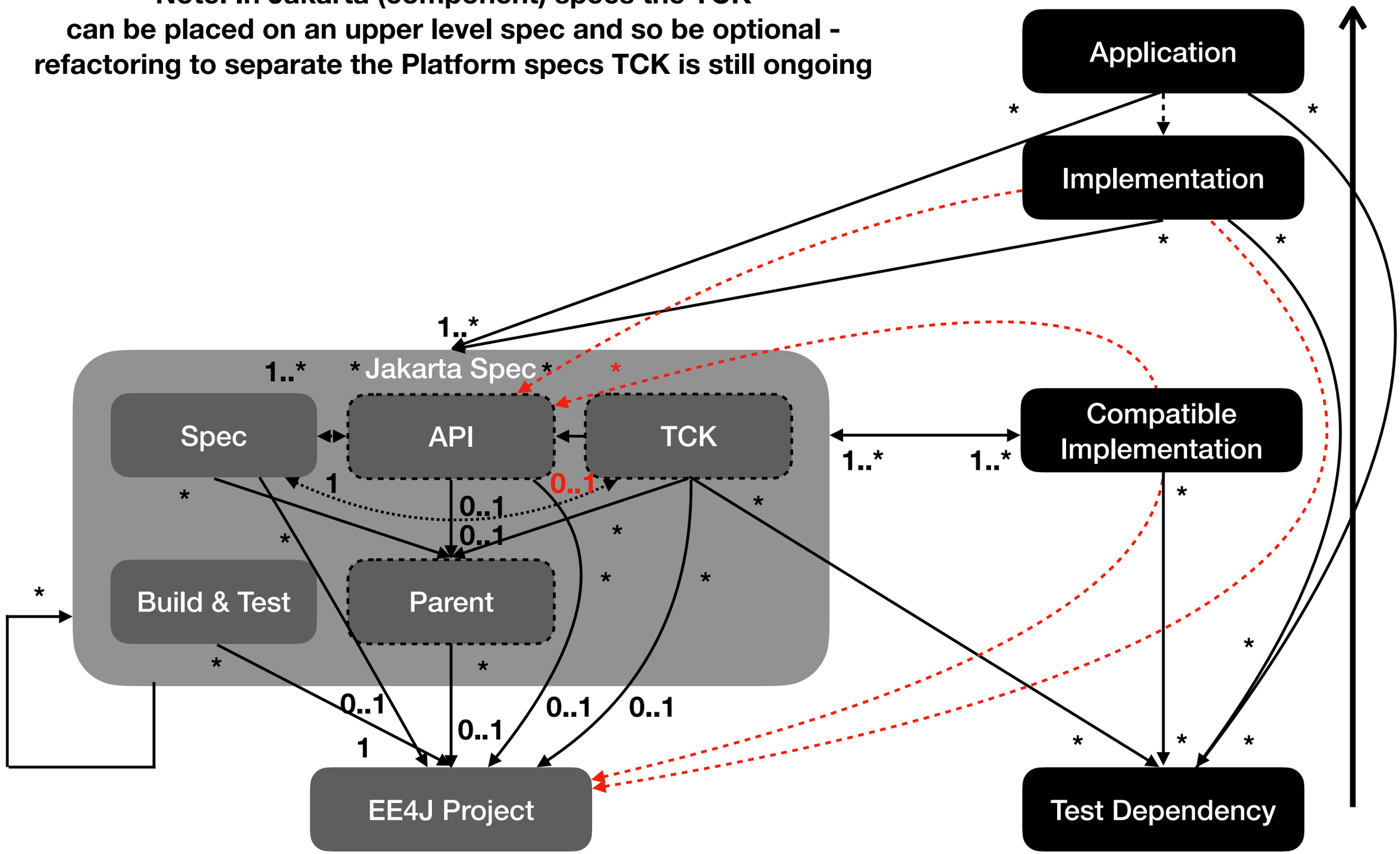


Jakarta EE Spec

- Issue
    - Need to define it’s own build and test dependencies
    - License may deviate from Jakarta EE Parent
    - Implementation license may deviate from spec (API re-/overwrite)
    - Dependencies may deviate in component spec parts - testing deviating environment!
    - Versions may deviate in component spec parts - testing deviating environment!
    - Artifact names may deviate in component spec parts
    - Namespaces must deviate in component spec parts (especially in TCK)
    - Implementation tests may use component spec Namespace
    - Dependencies may deviate in spec dependency graph - testing deviating environment!
- Solution
    - Use unified Jakarta EE Parent
    - Add missing licenses to Jakarta EE Parent or create alternative Jakarta EE Parent (part)?
    - Use mono repository with parent that defines versions for components and dependencies
    - TCK-BOM prevents test dependencies on non-TCKs
    - Use unified naming for specs and components
    - Keep all dependencies in sync for the spec dependency graph

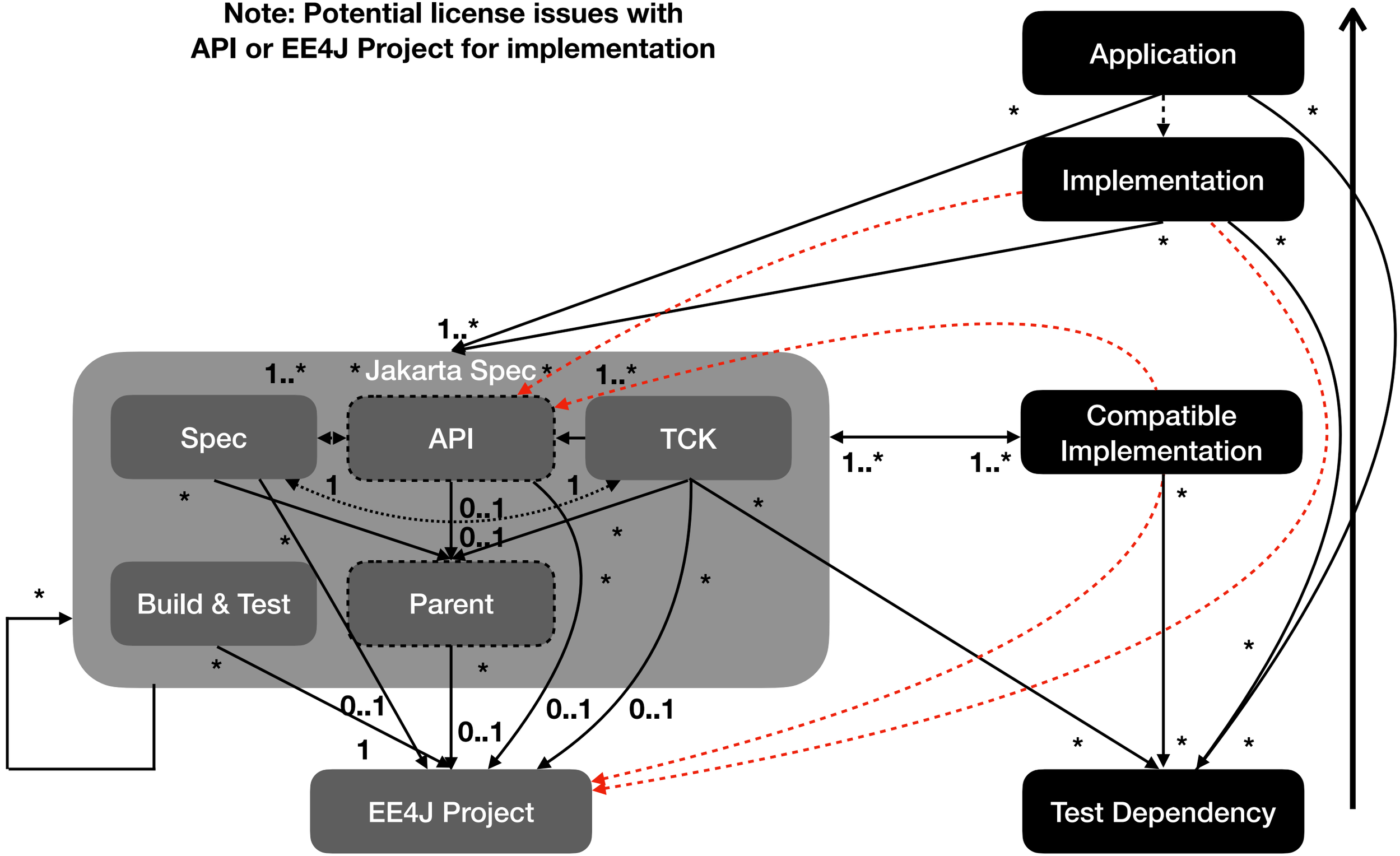
Jakarta EE Spec Detail

Note: In Jakarta (component) specs the TCK can be placed on an upper level spec and so be optional - refactoring to separate the Platform specs TCK is still ongoing



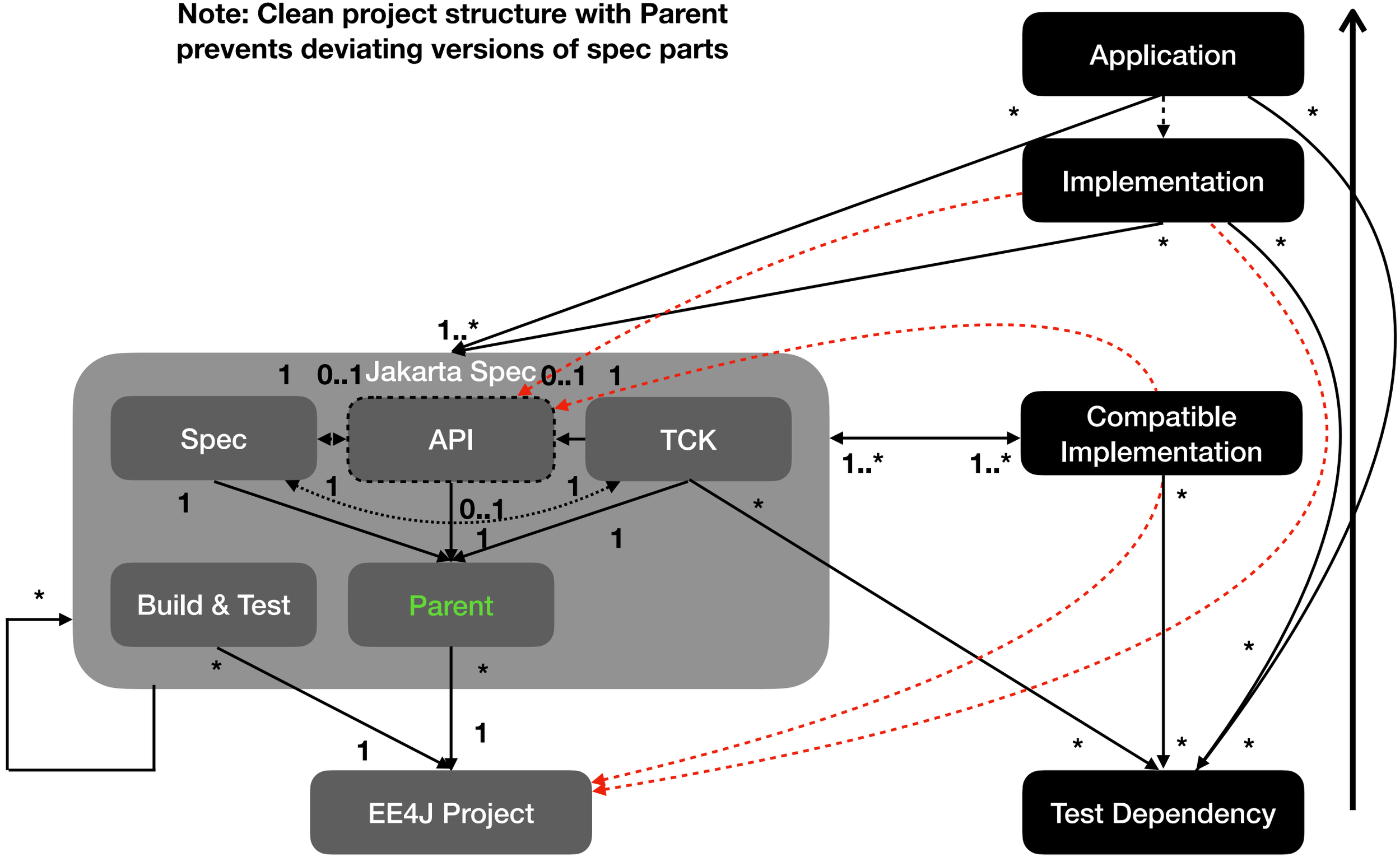
Jakarta EE Spec Detail

Note: Potential license issues with
API or EE4J Project for implementation



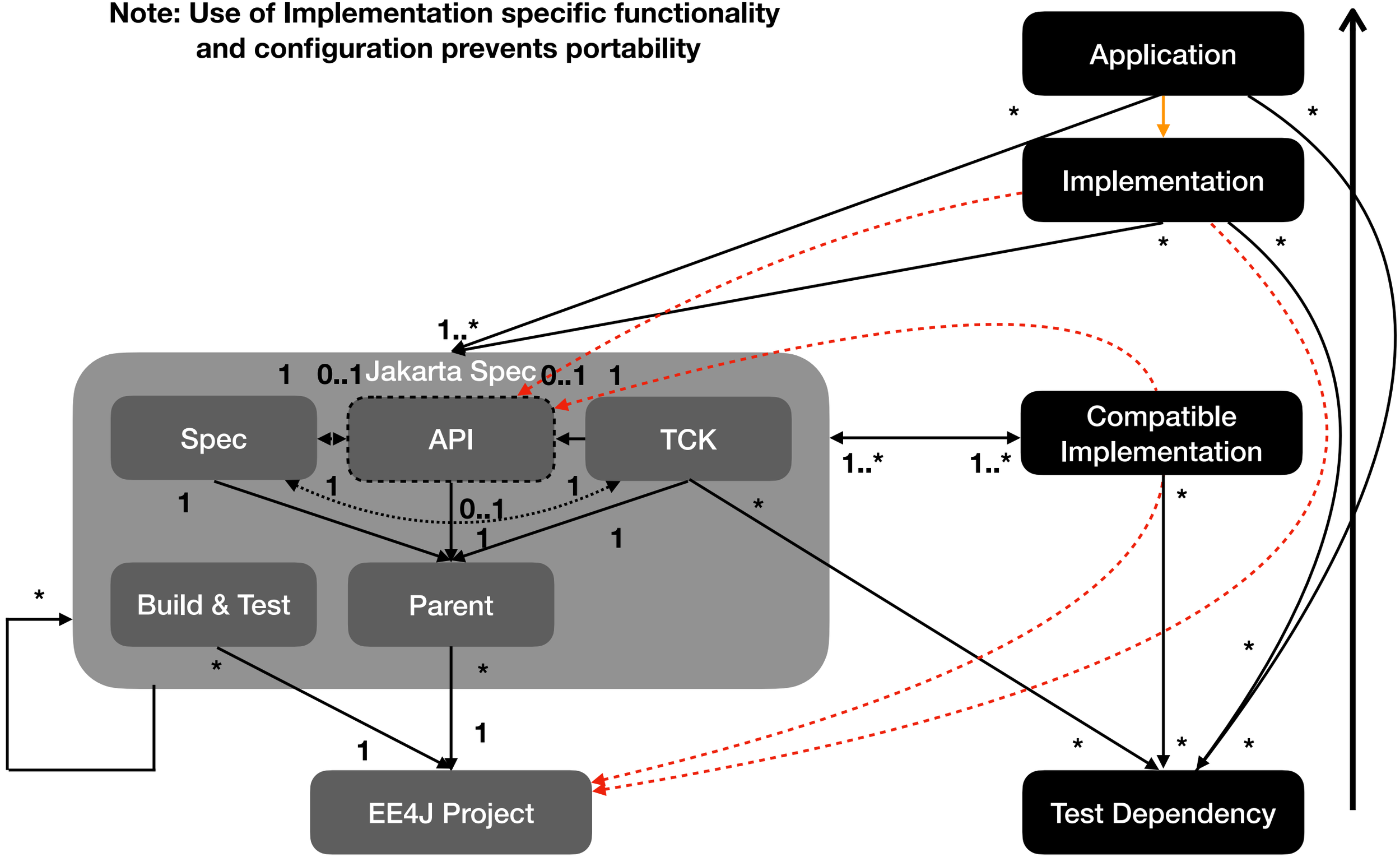
Jakarta EE Spec Detail

Note: Clean project structure with Parent prevents deviating versions of spec parts



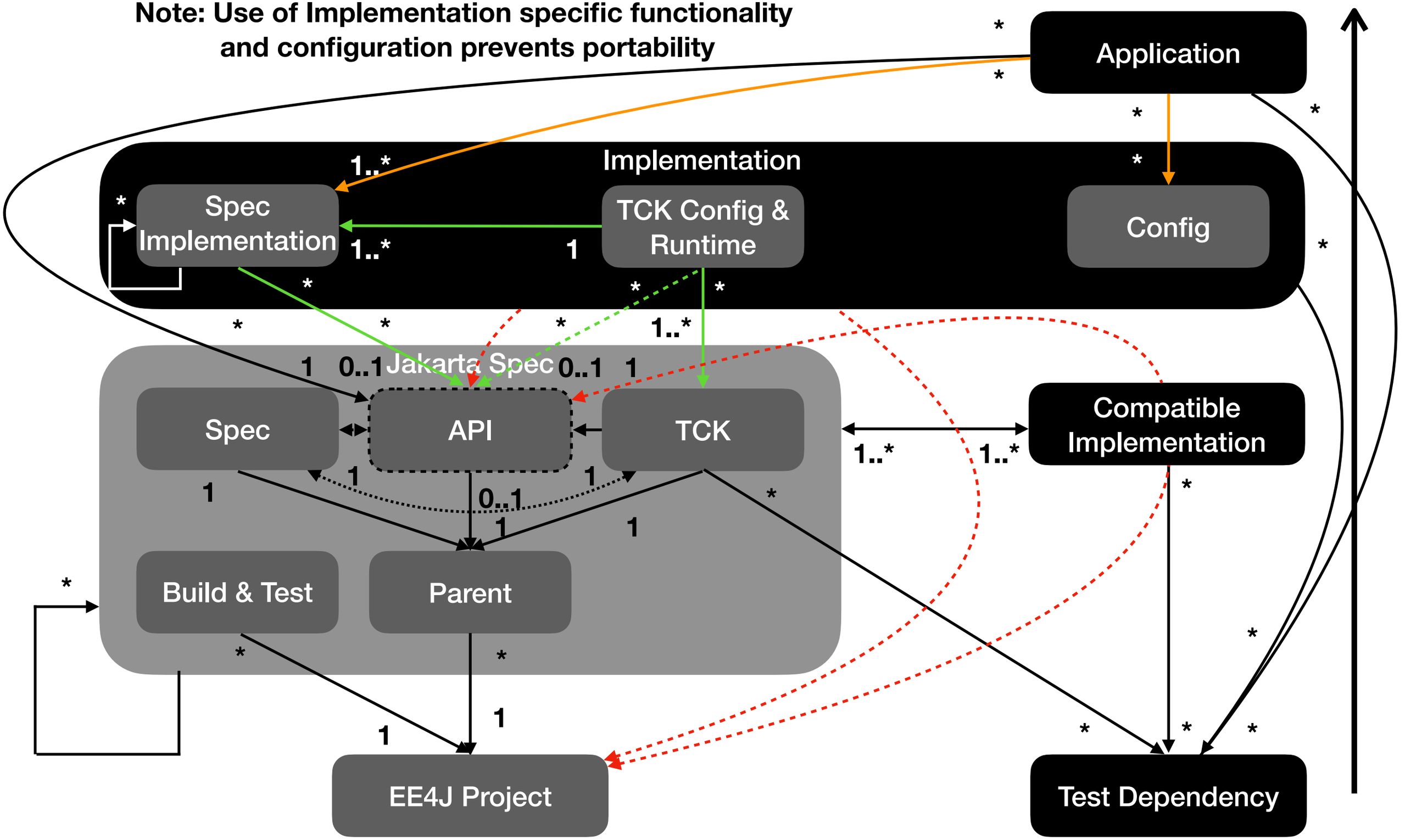
Jakarta EE Spec Detail

Note: Use of Implementation specific functionality and configuration prevents portability



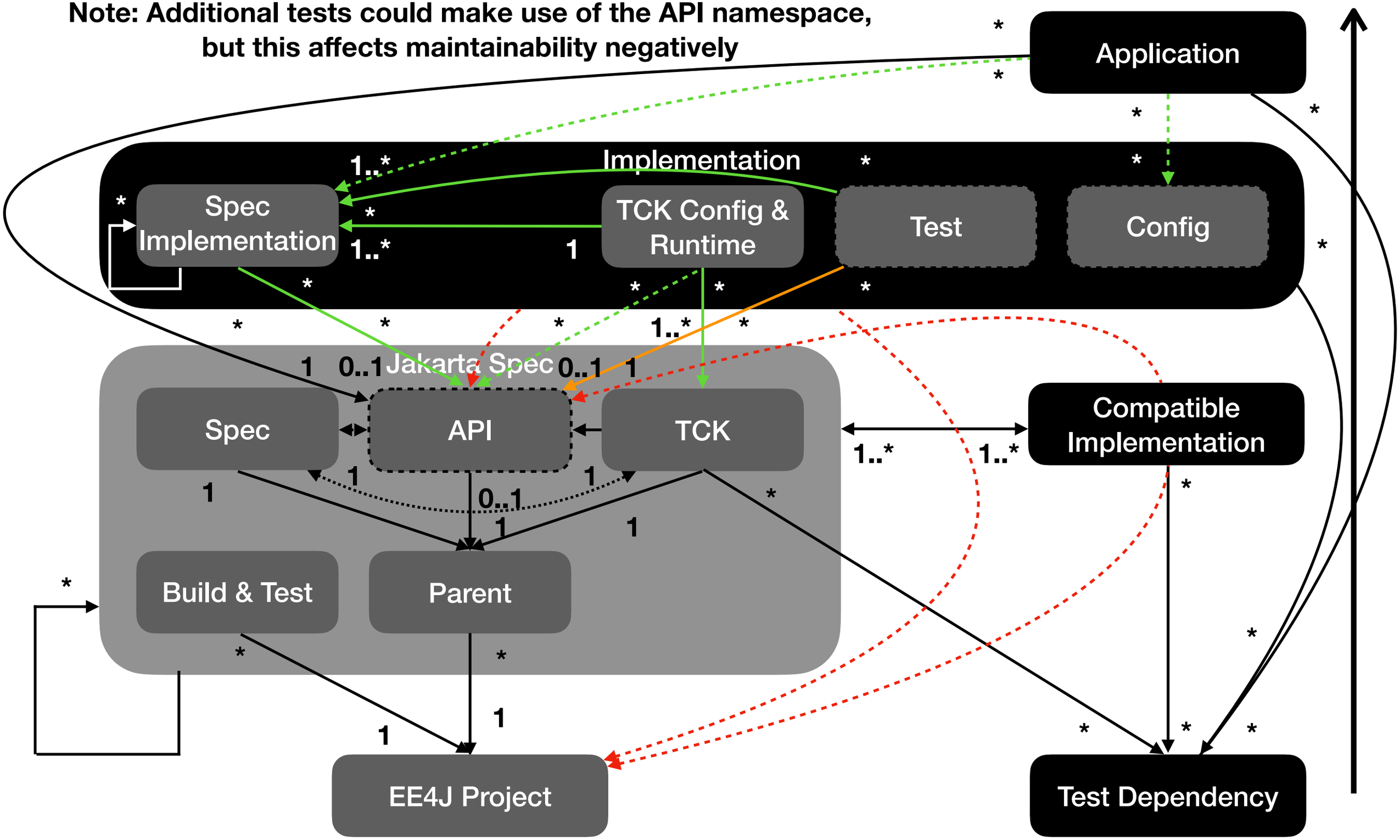
Jakarta EE Spec Detail

Note: Use of Implementation specific functionality and configuration prevents portability



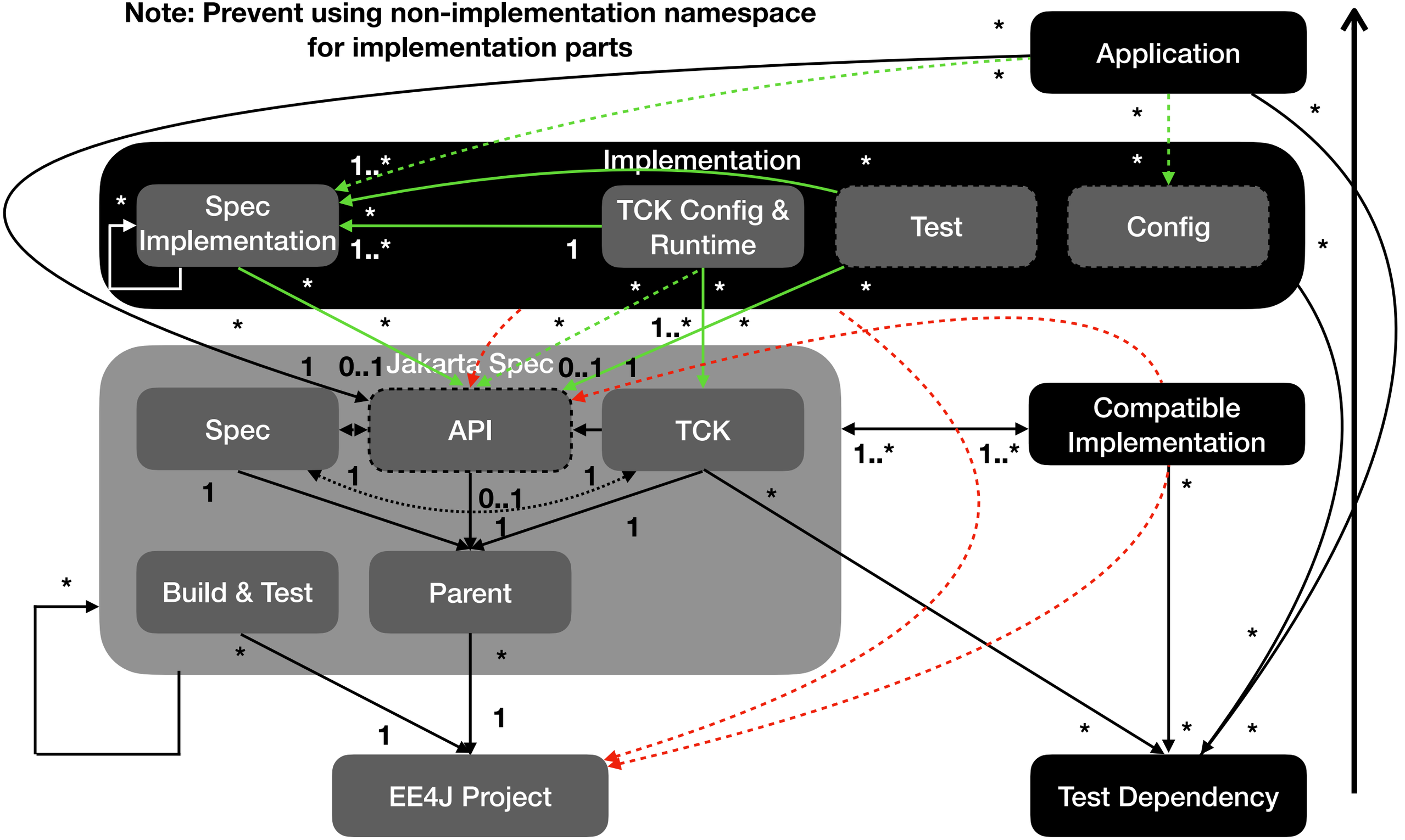
Jakarta EE Spec Detail

Note: Additional tests could make use of the API namespace, but this affects maintainability negatively



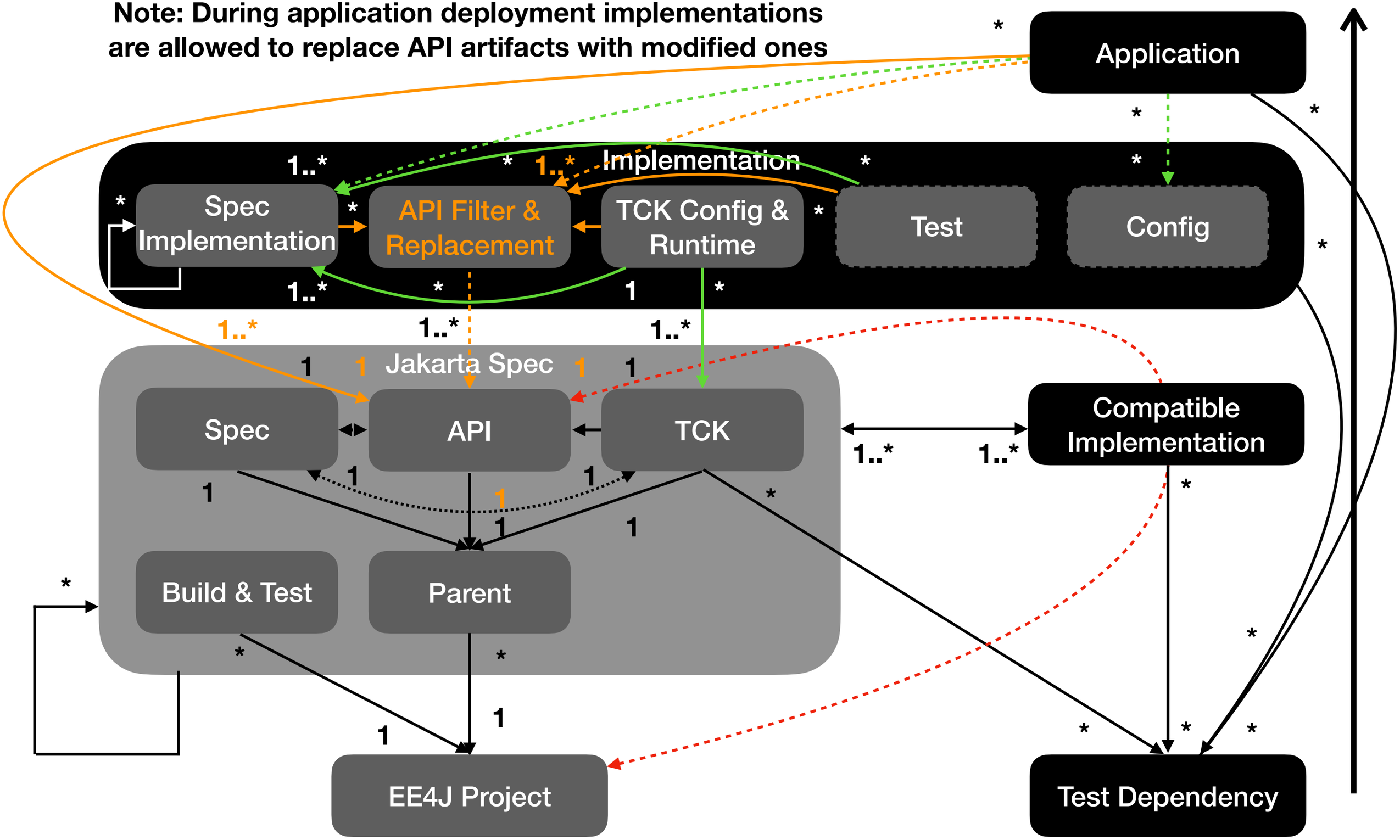
Jakarta EE Spec Detail

Note: Prevent using non-implementation namespace for implementation parts



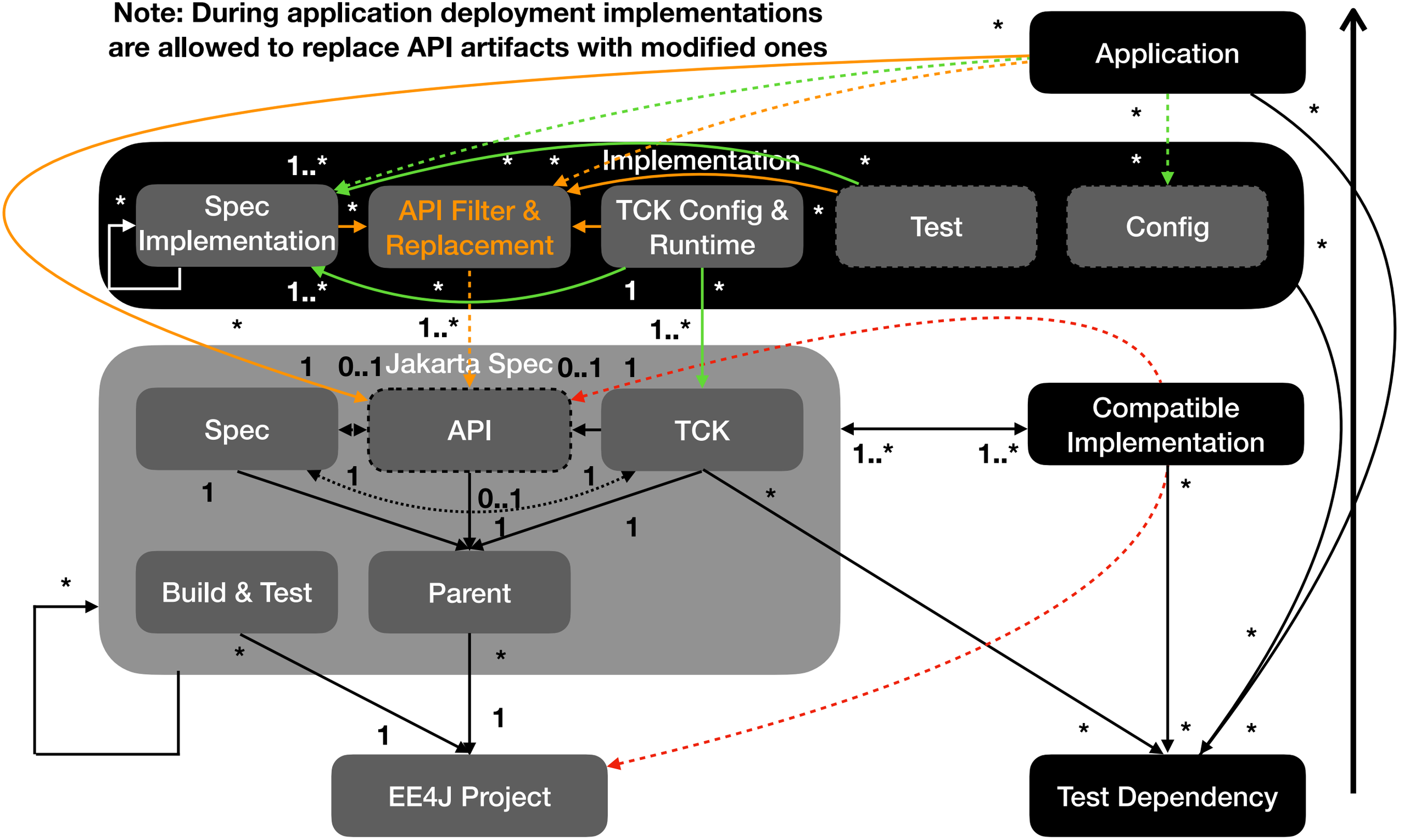
Jakarta EE Spec Detail

Note: During application deployment implementations are allowed to replace API artifacts with modified ones



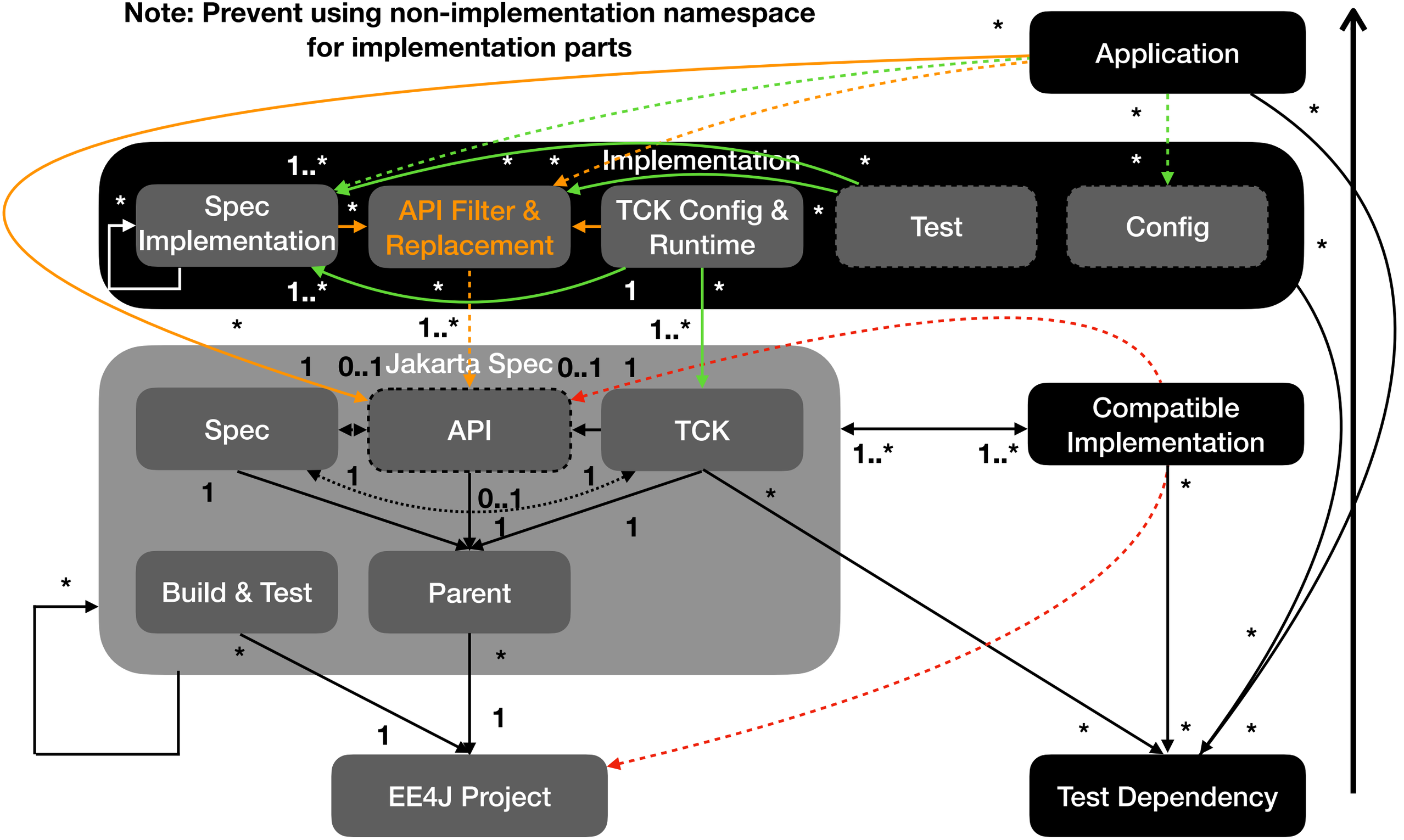
Jakarta EE Spec Detail

Note: During application deployment implementations are allowed to replace API artifacts with modified ones



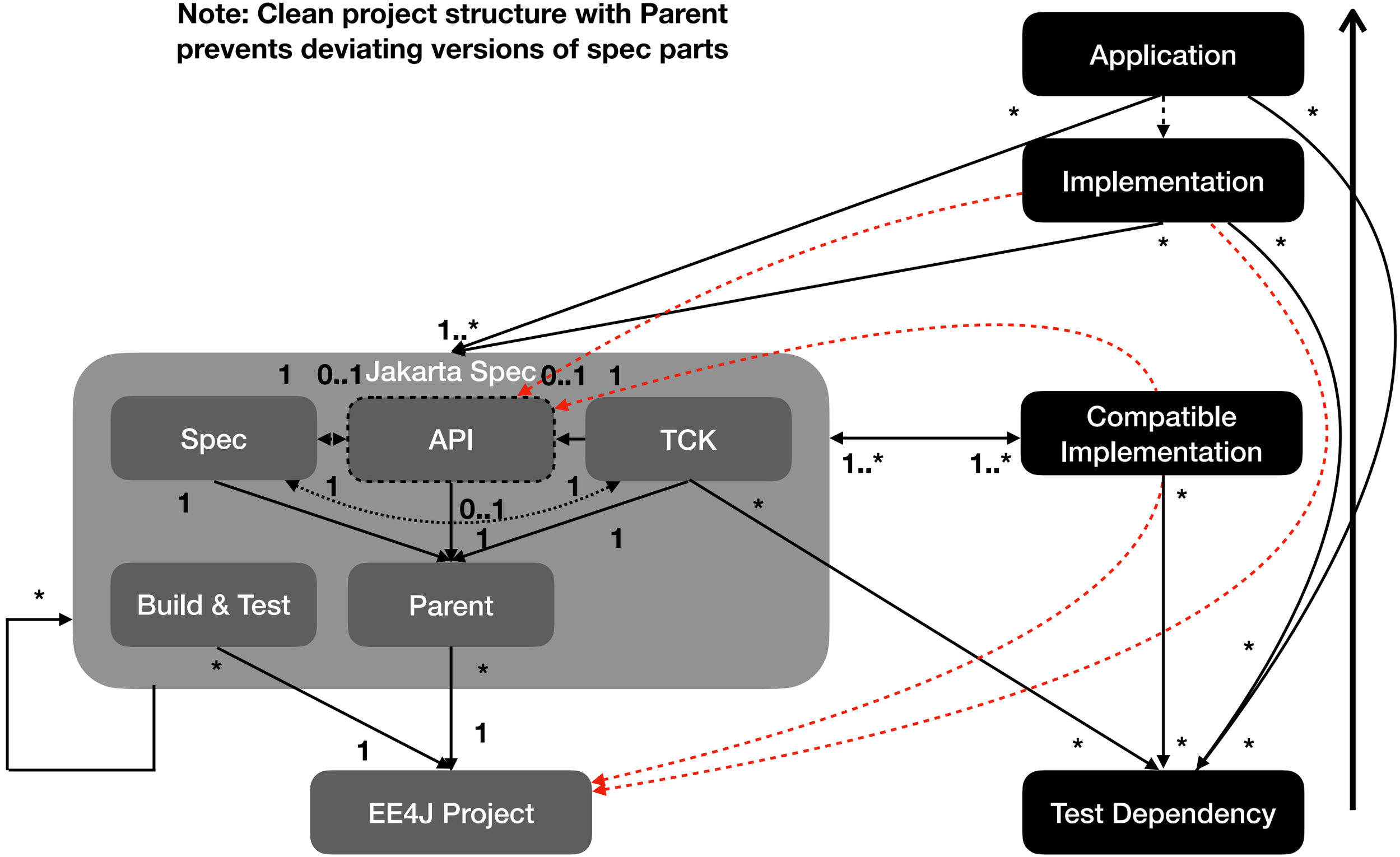
Jakarta EE Spec Detail

Note: Prevent using non-implementation namespace for implementation parts



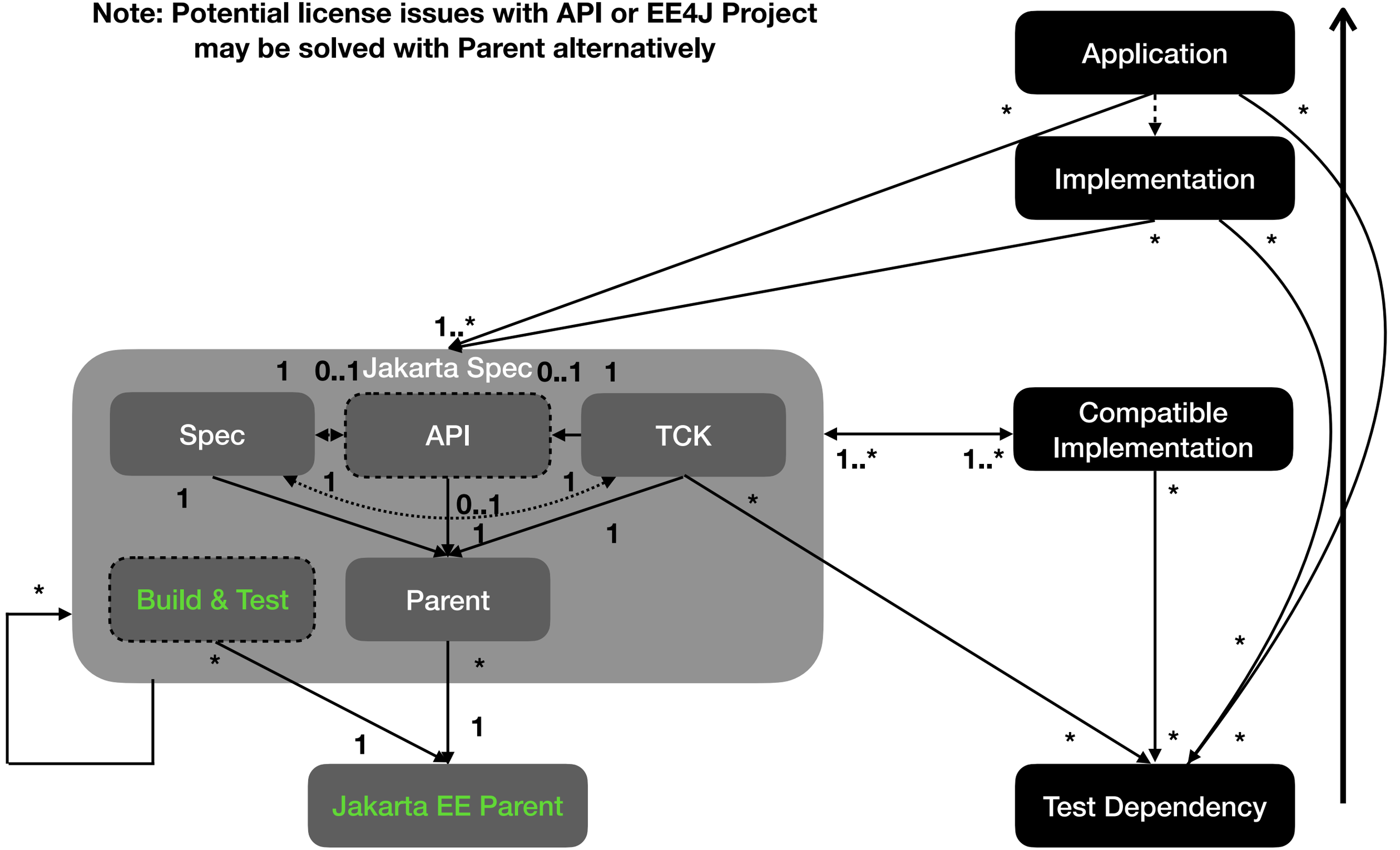
Jakarta EE Spec Detail

Note: Clean project structure with Parent prevents deviating versions of spec parts



Jakarta EE Spec Detail

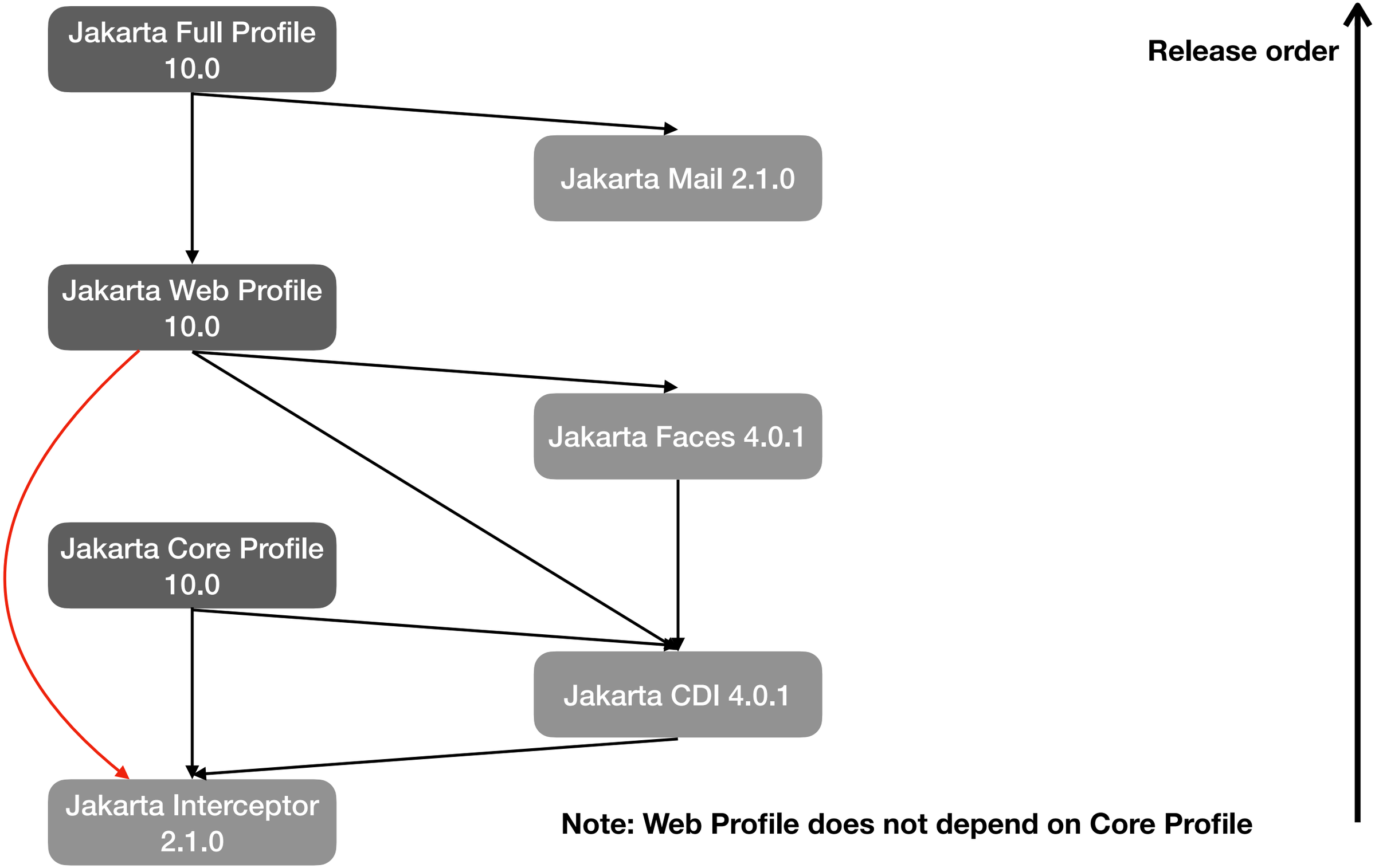
Note: Potential license issues with API or EE4J Project may be solved with Parent alternatively



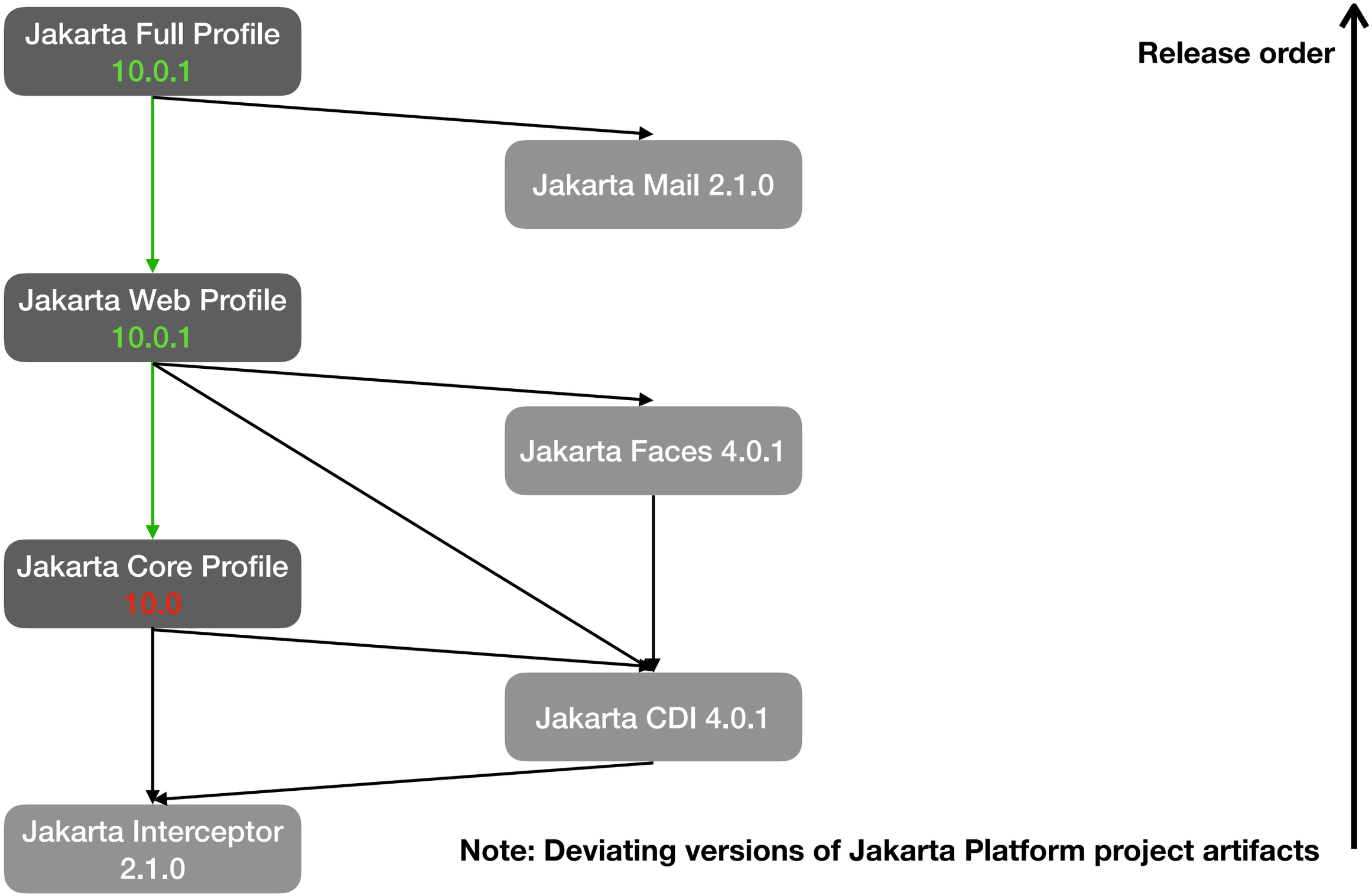
Jakarta EE Platform

- Issue
  - Web Profile does not depend on Core Profile
  - Intention to have independent Core Profile releases
- Solution
  - Web Profile should depend on Core Profile
  - All Jakarta EE Platform releases should be in sync (regarding dependencies and version)

Jakarta EE Platform

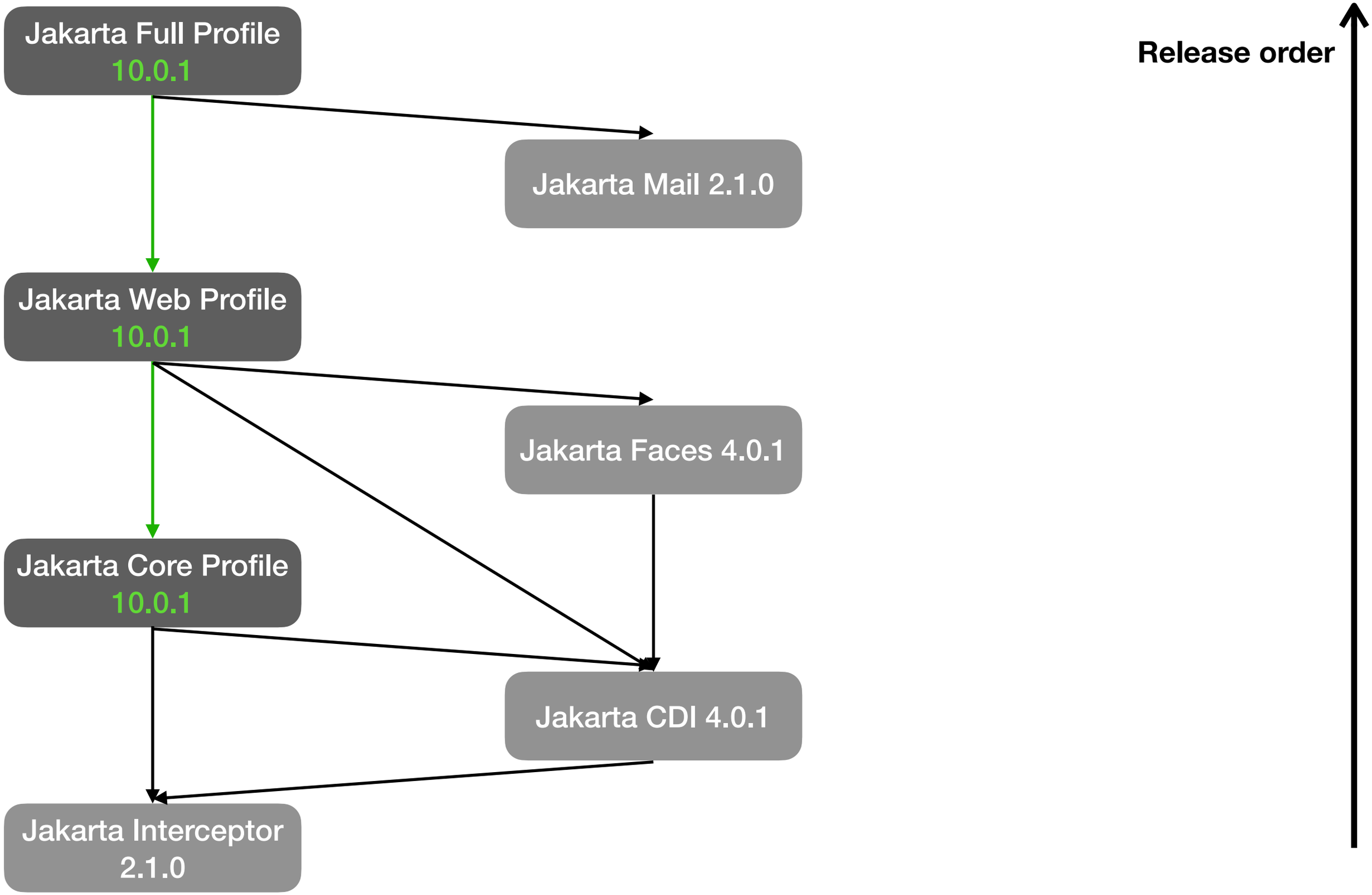


Jakarta EE Platform



Note: Deviating versions of Jakarta Platform project artifacts

Jakarta EE Platform



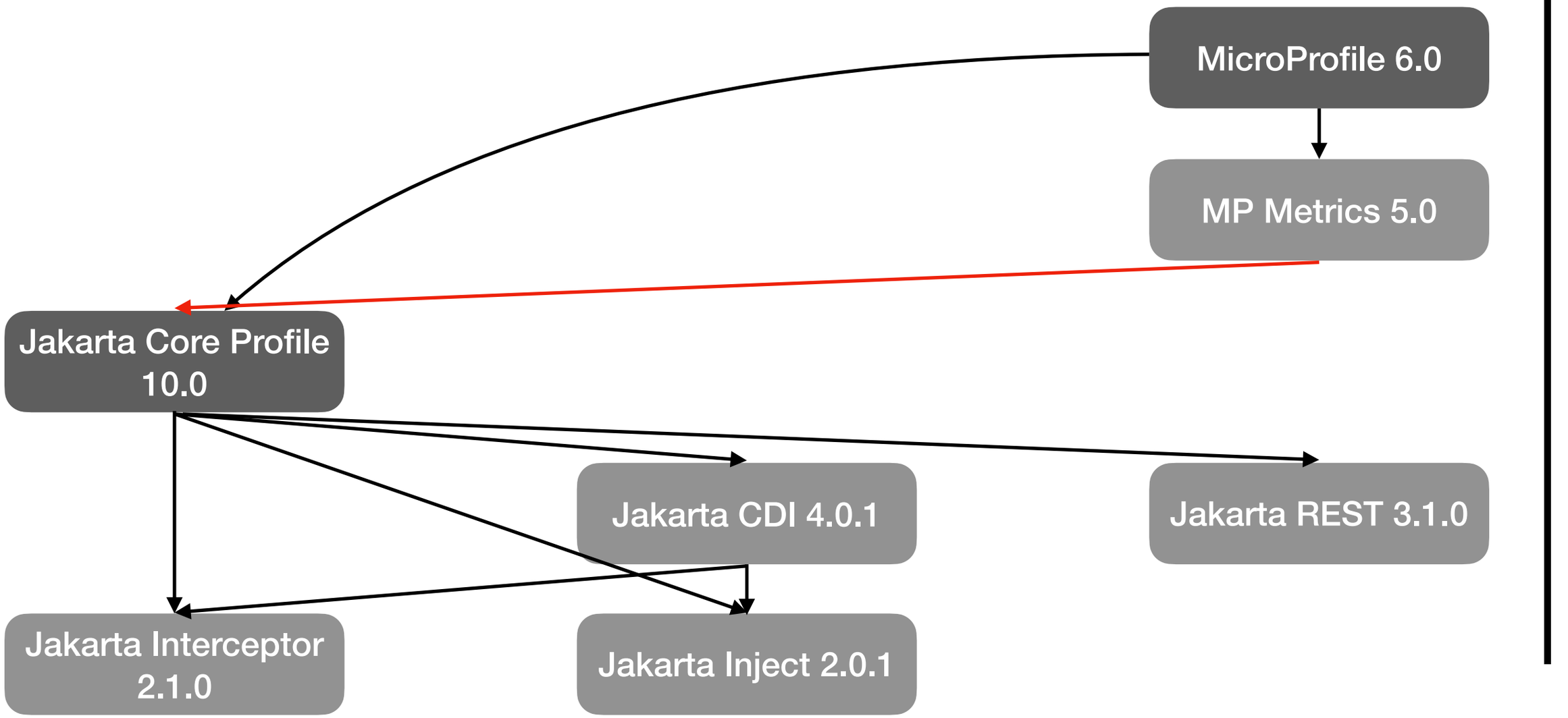
MicroProfile Metrics

- Issue
  - Component spec depends on umbrella spec
- Solution
  - MicroProfile should depend on component specs only

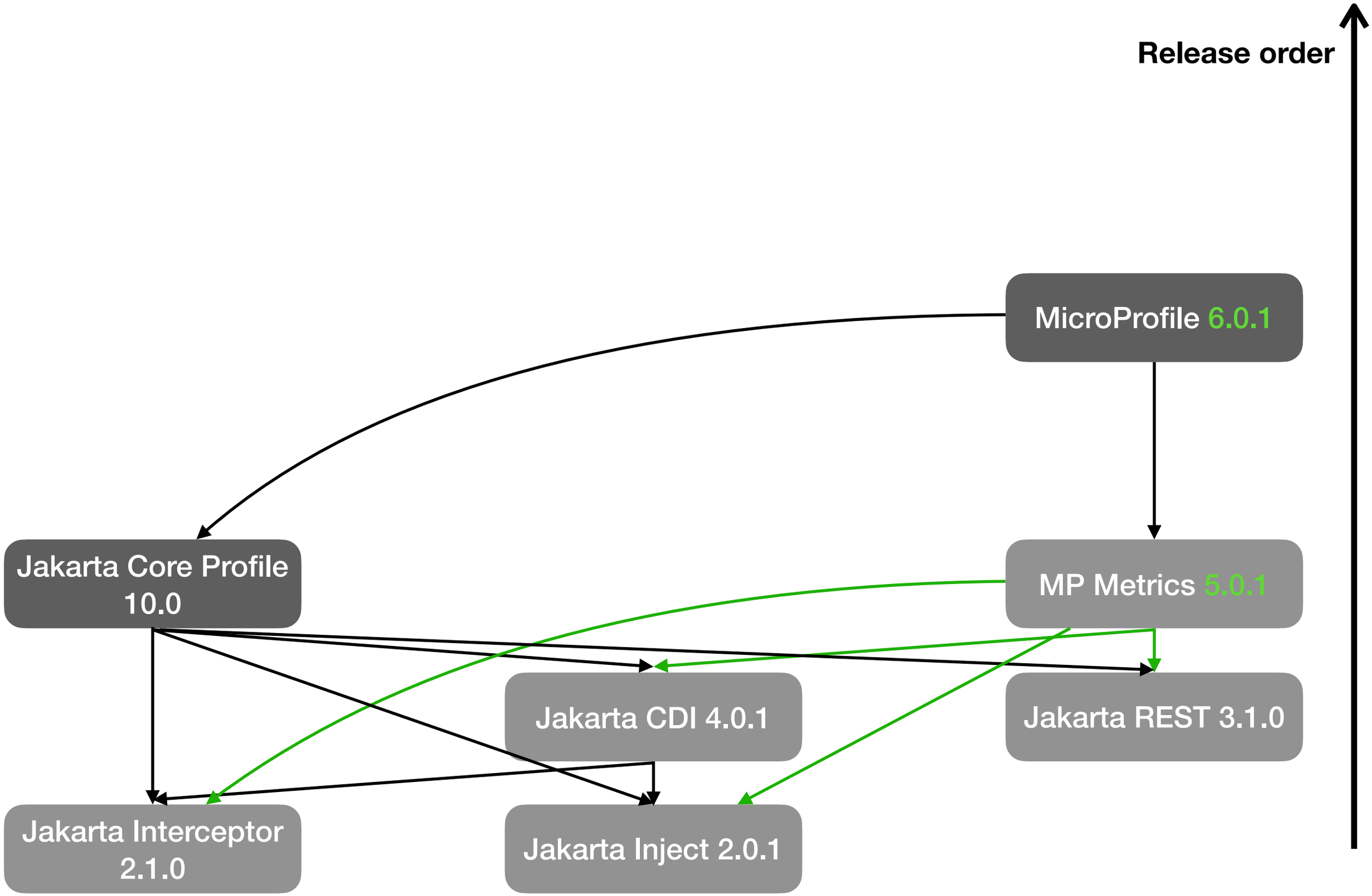
MicroProfile Metrics

Note: Component spec references umbrella spec

Release order



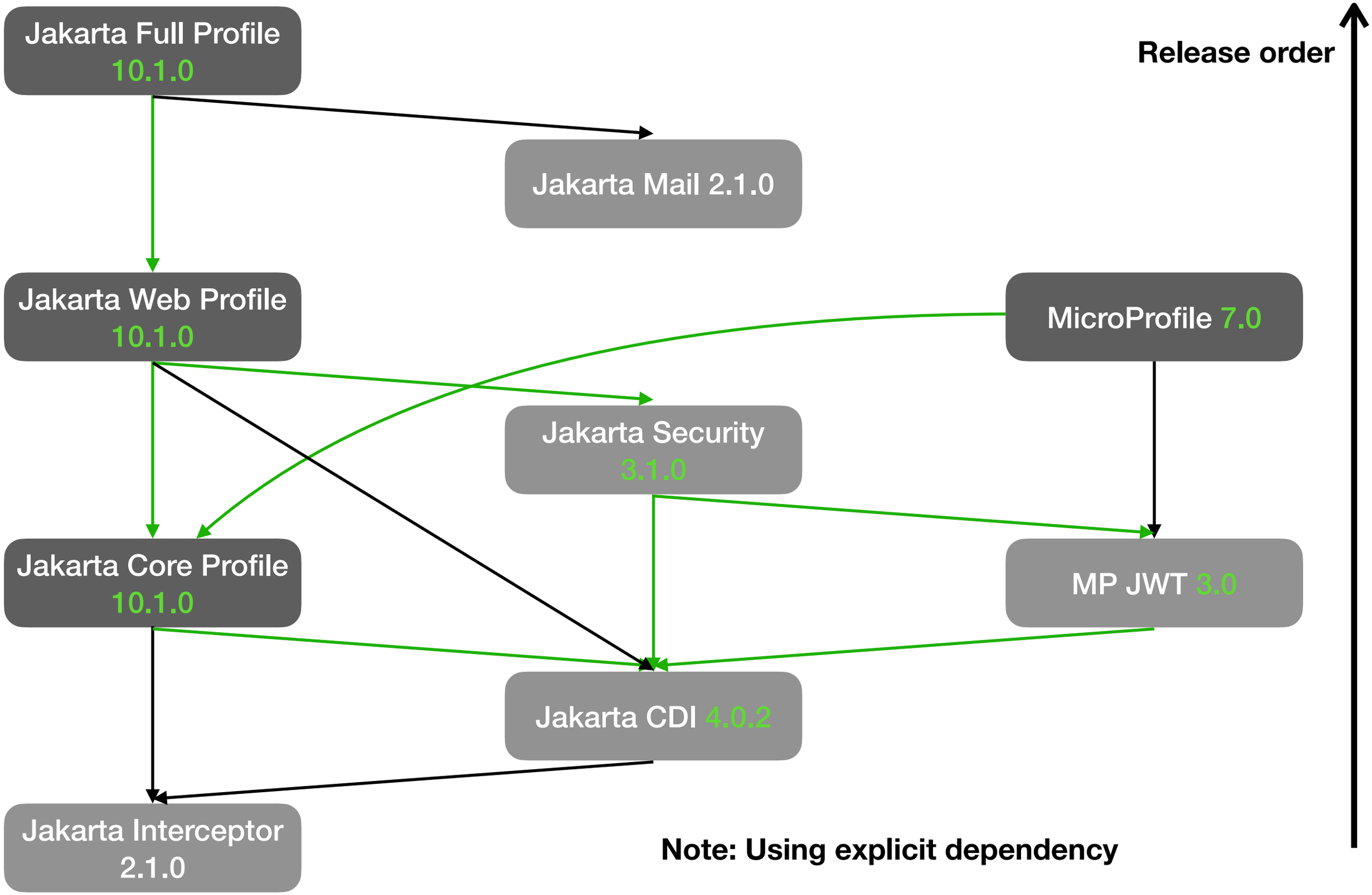
MicroProfile Metrics



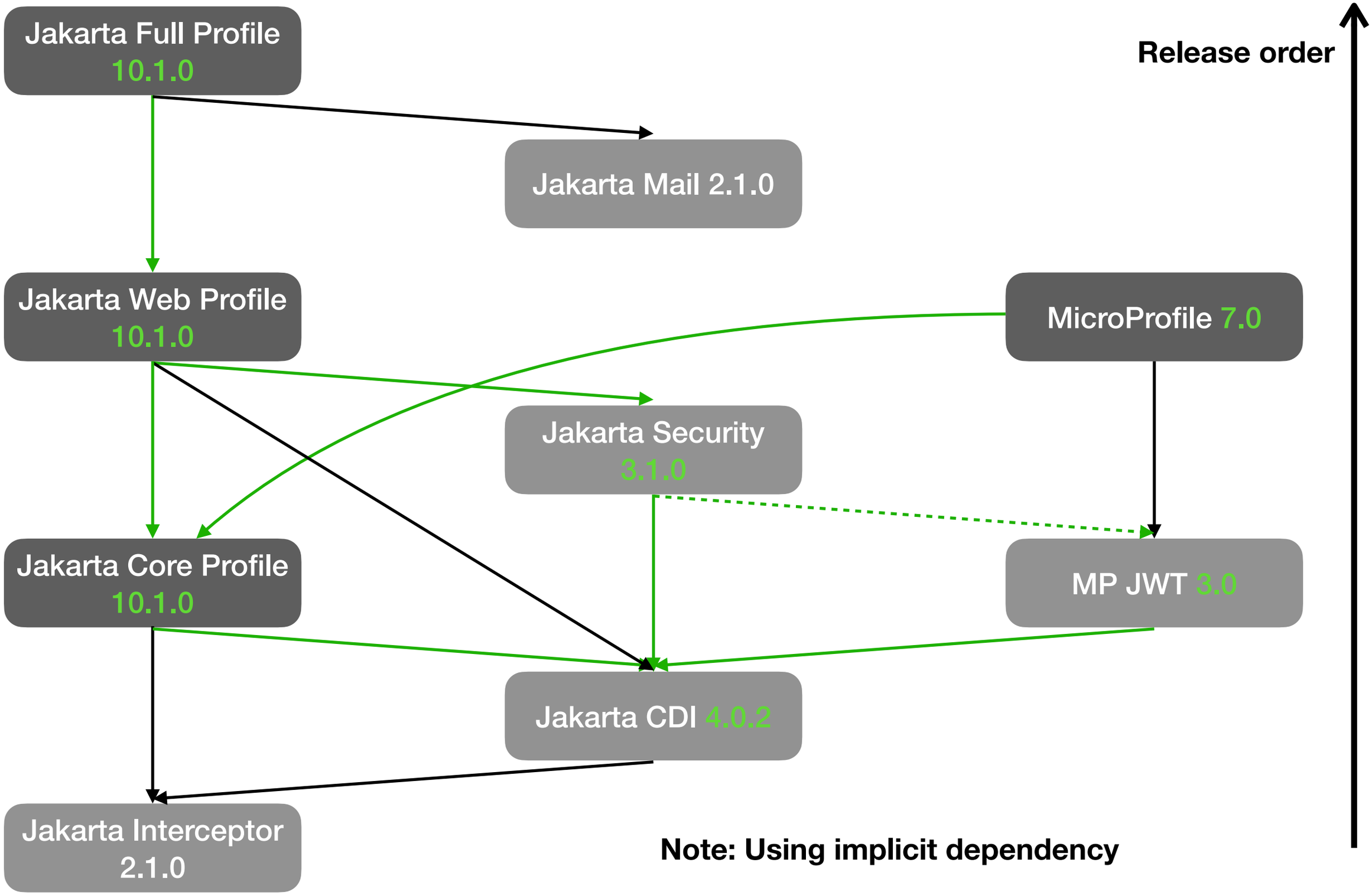
Jakarta Security & MP JWT

- Issue
  - Jakarta Security wants to include or consume (MP) JWT
  - Different organisations and umbrella specs involved
    - Deviation in dependencies and release cycles
- Solution
  - Definition of explicit (strong) dependency
  - Definition of implicit (weak) dependency

Jakarta Security & MP JWT



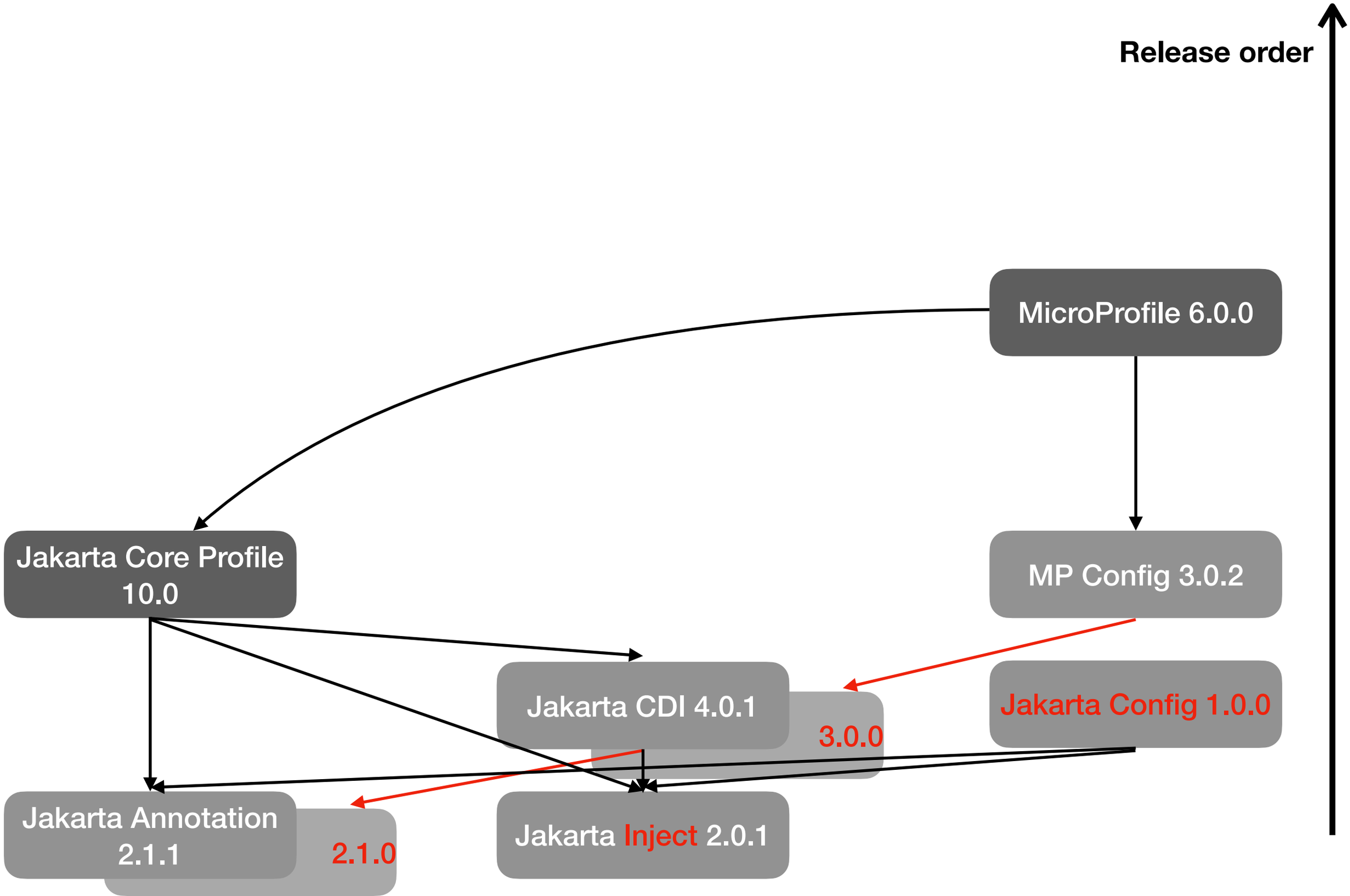
Jakarta Security & MP JWT



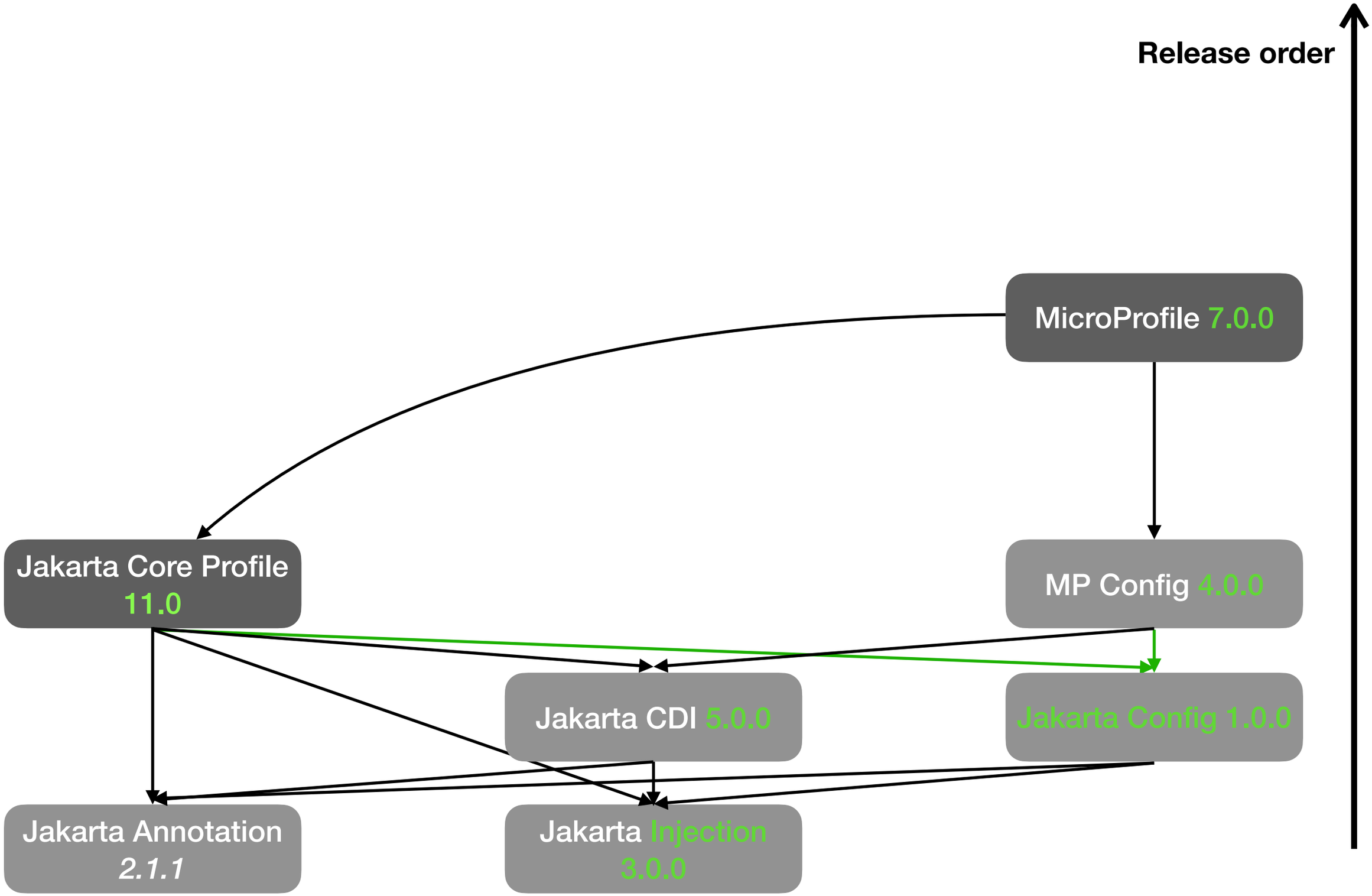
MP Config & Jakarta Config

- Issue
  - First approach to move spec failed (spec modified during transition)
    - Dependency mismatch
- Solution
  - Second MVP approach (defining minimal functionality)
  - Jakarta Config should become real subset of MP Config
  - Renaming of Jakarta Inject(ion)?

MP Config & Jakarta Config



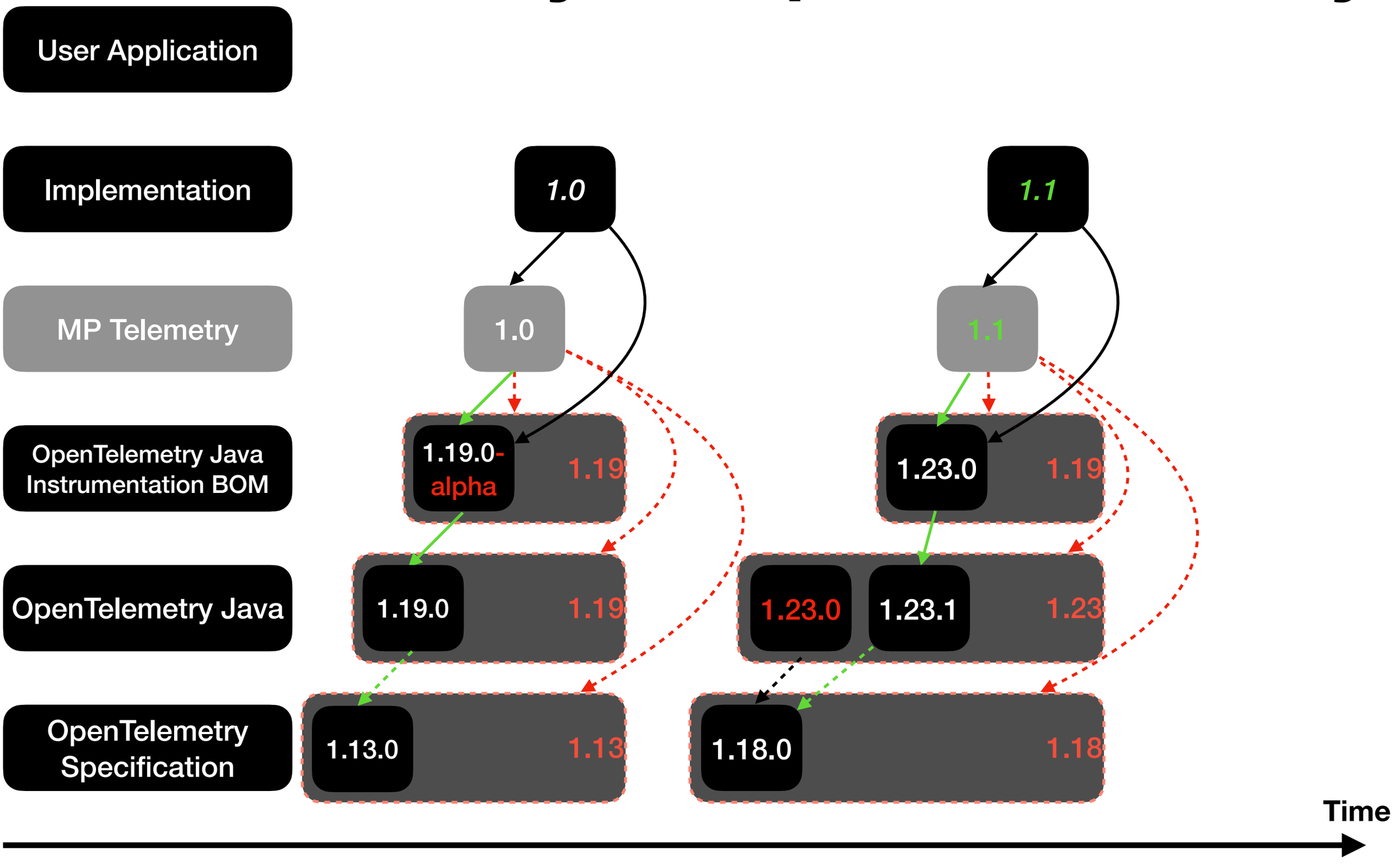
MP Config & Jakarta Config



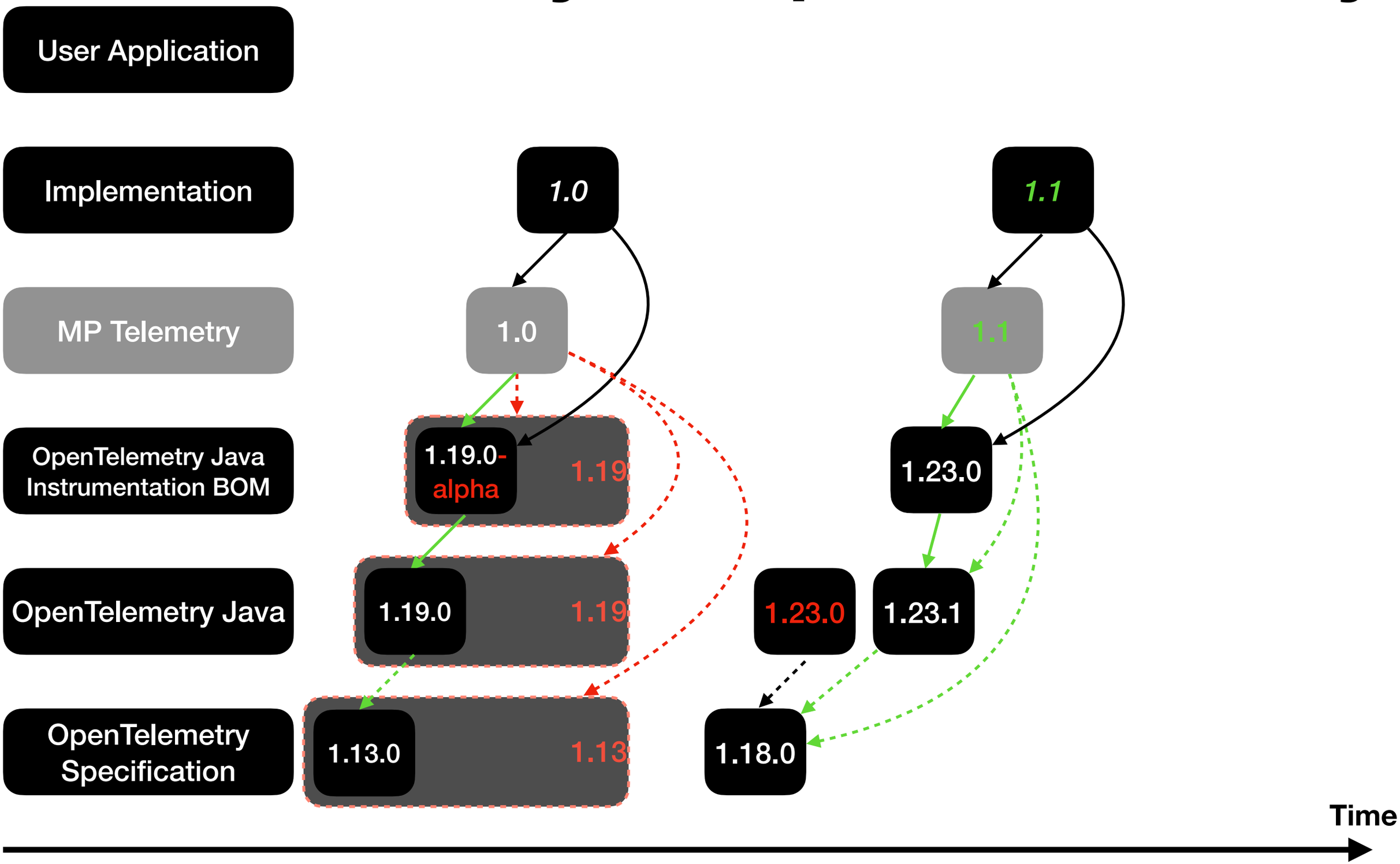
MP Telemetry & OpenTelemetry

- Issue
  - Do not depend on umbrella spec directly
  - Using (again) separate definitions of spec versions in the spec document and API/TCK - manual maintenance needed
  - Using unspecific (non existent) version definitions in Spec document - allows usage of broken versions knowingly!
    - opentelemetry-java and opentelemtry-java-instrumentation artifacts need to use the same version number
- Solution
  - Use component spec dependencies only
  - Use consistent version information in all spec artifacts
  - Define separate versions for all dependencies to be able to select deviating versions, when necessary
  - Define root MP Telemetry spec document

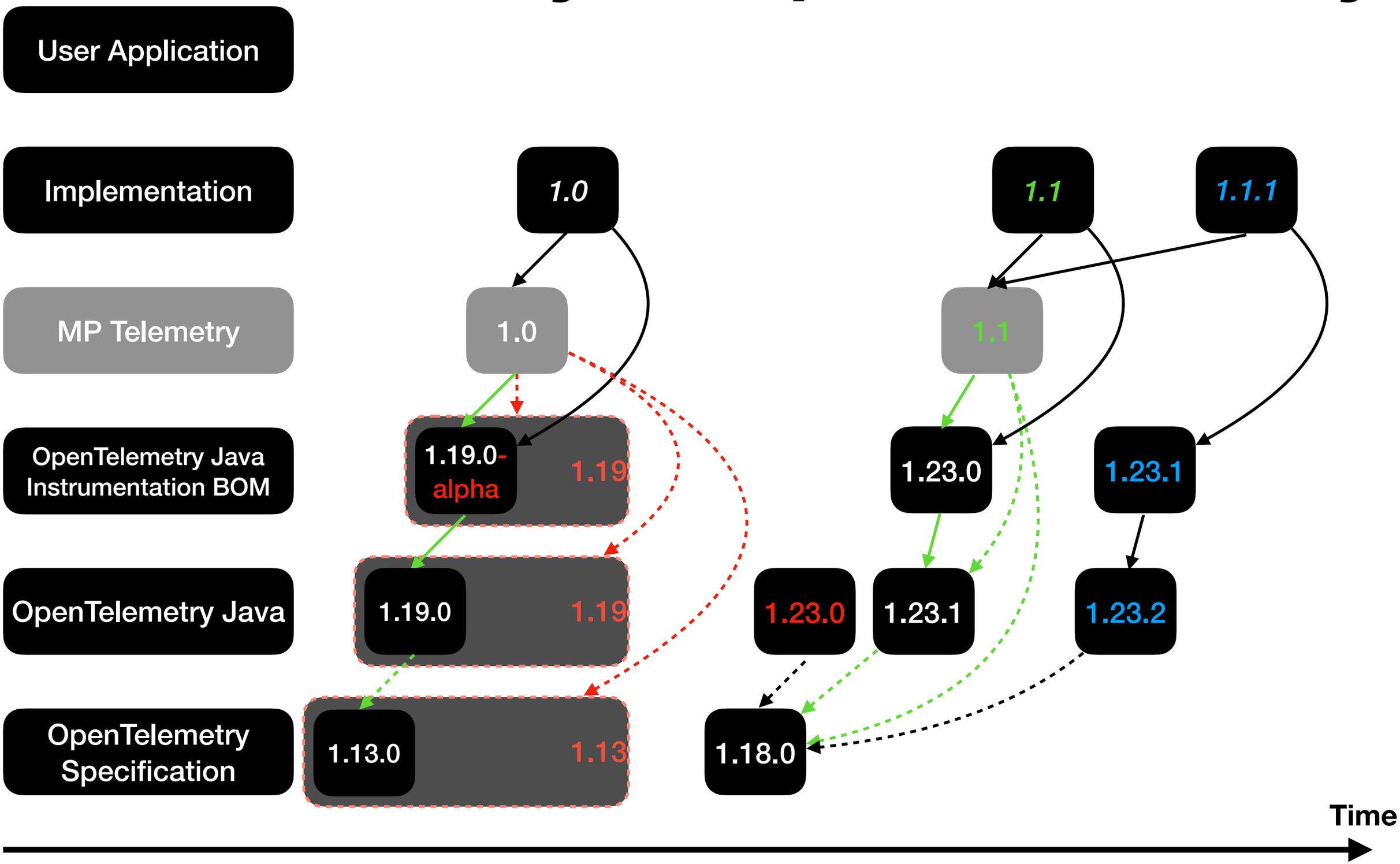
MP Telemetry & OpenTelemetry



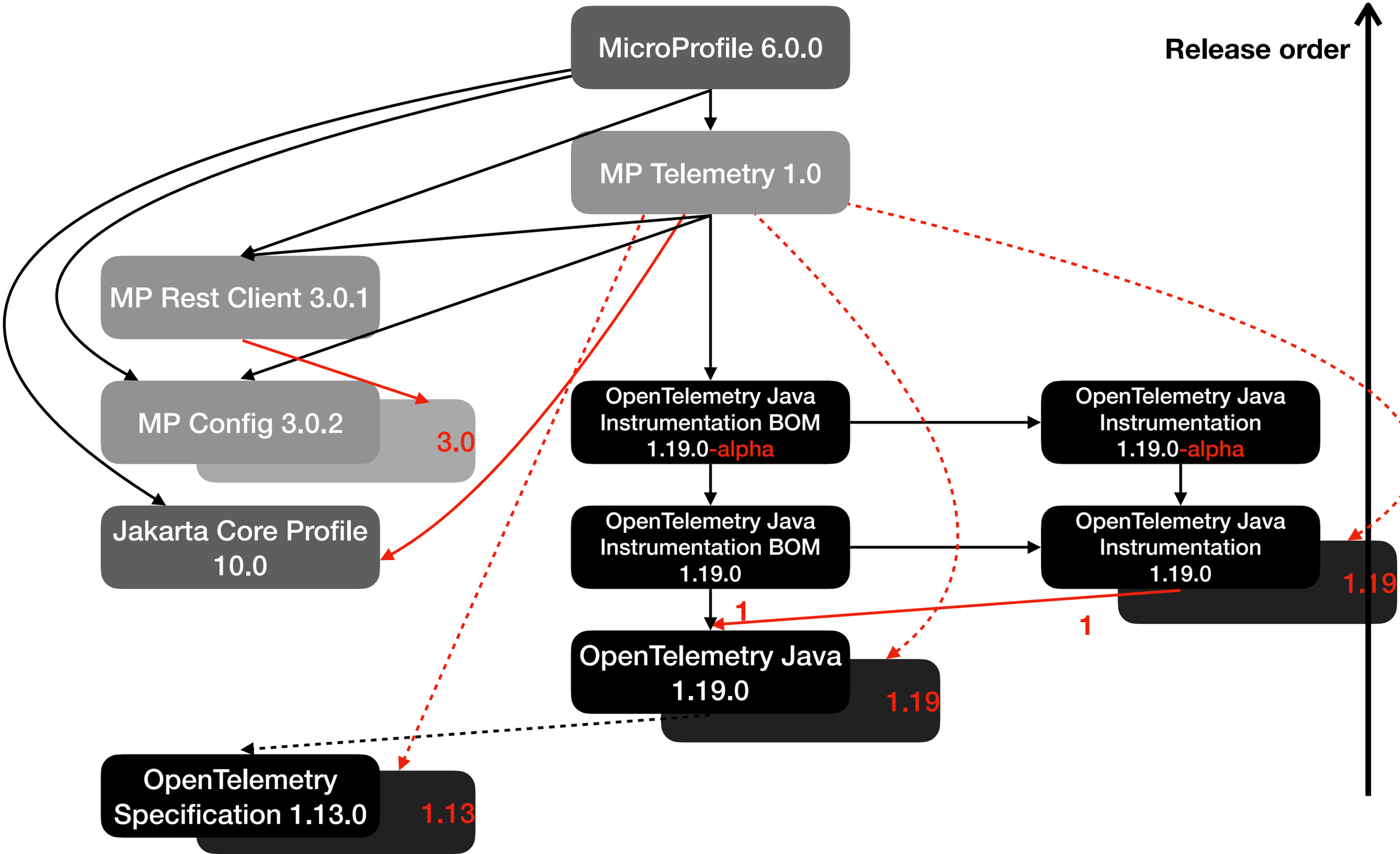
MP Telemetry & OpenTelemetry



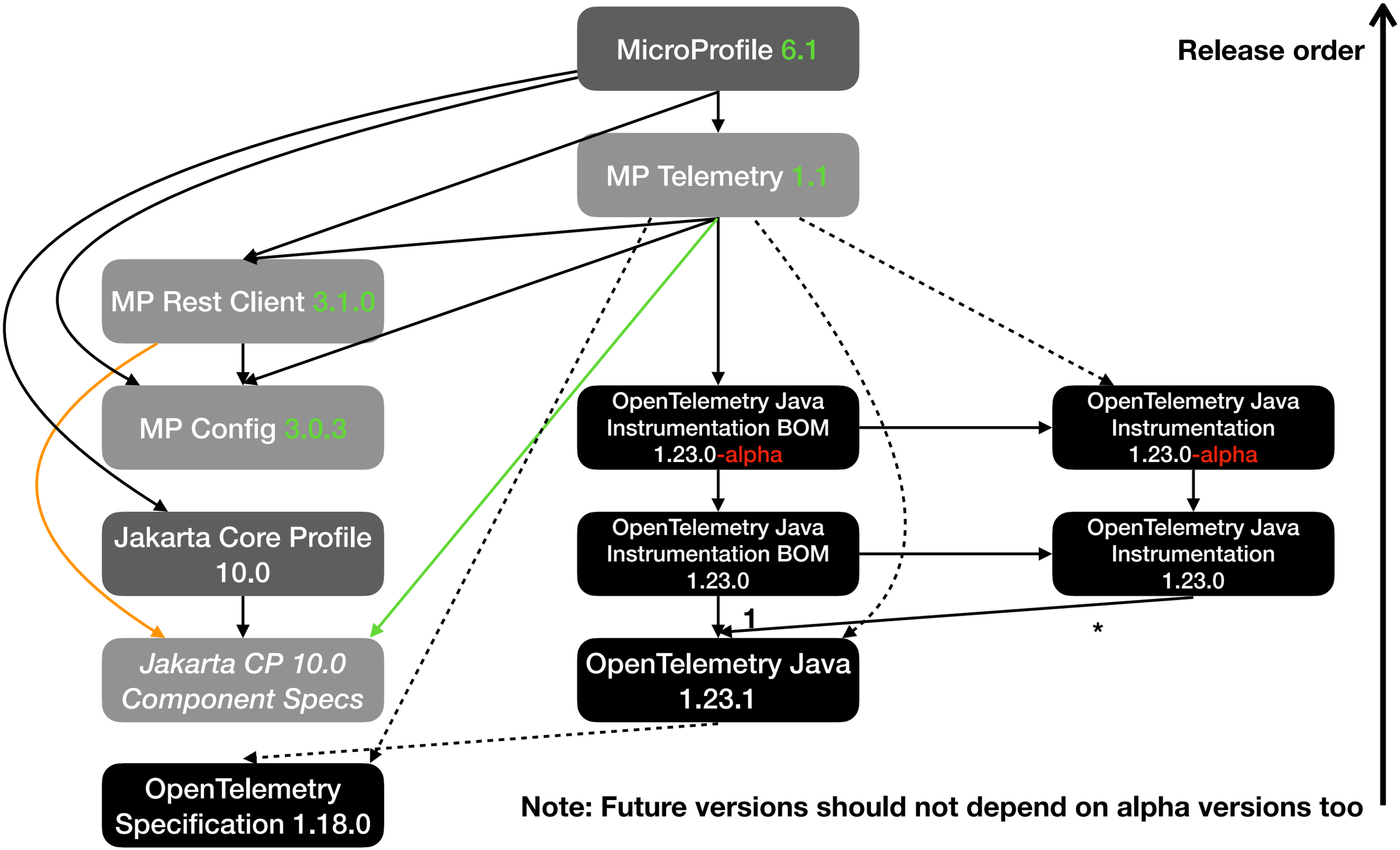
MP Telemetry & OpenTelemetry



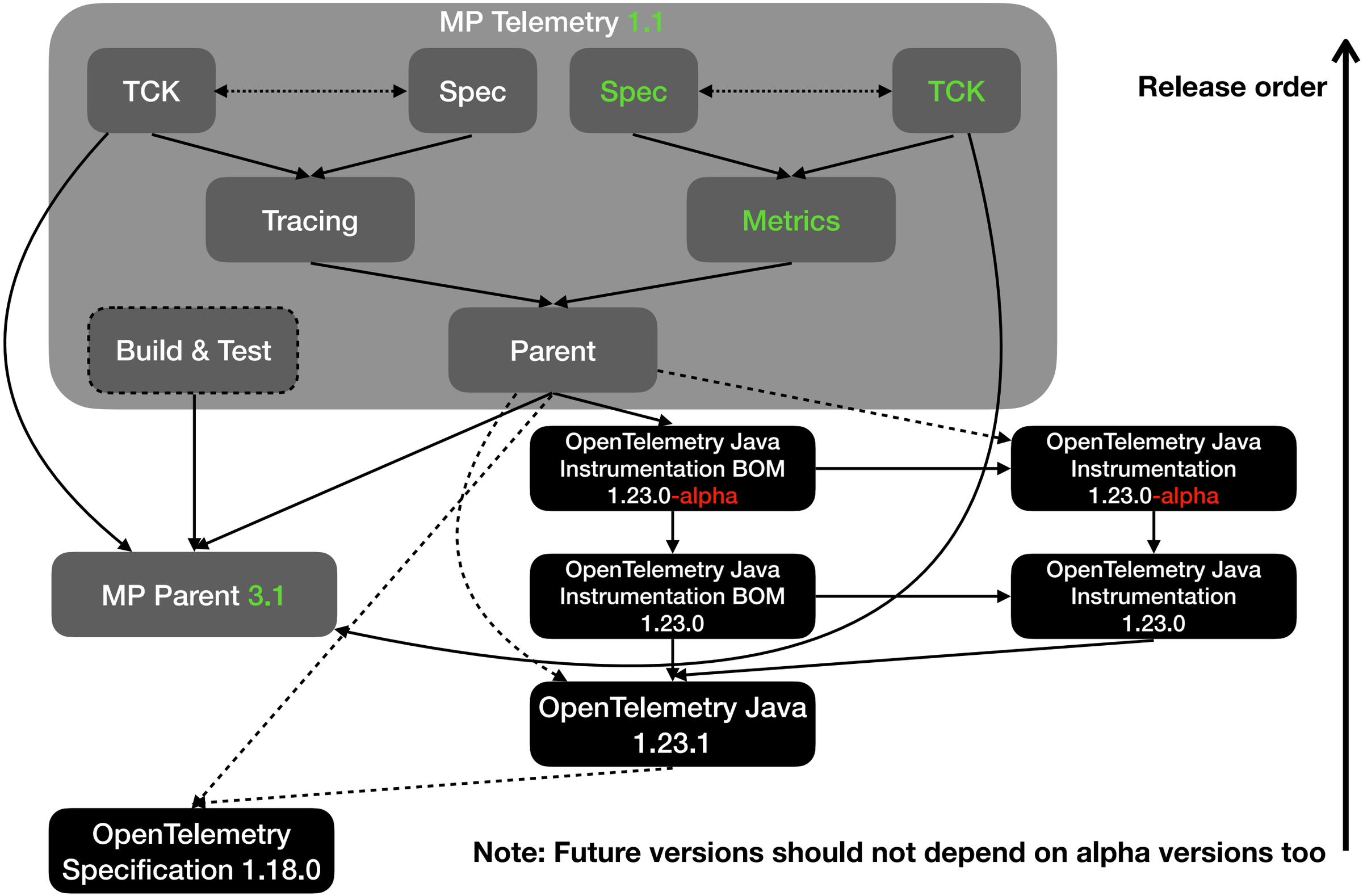
MP Telemetry & OpenTelemetry



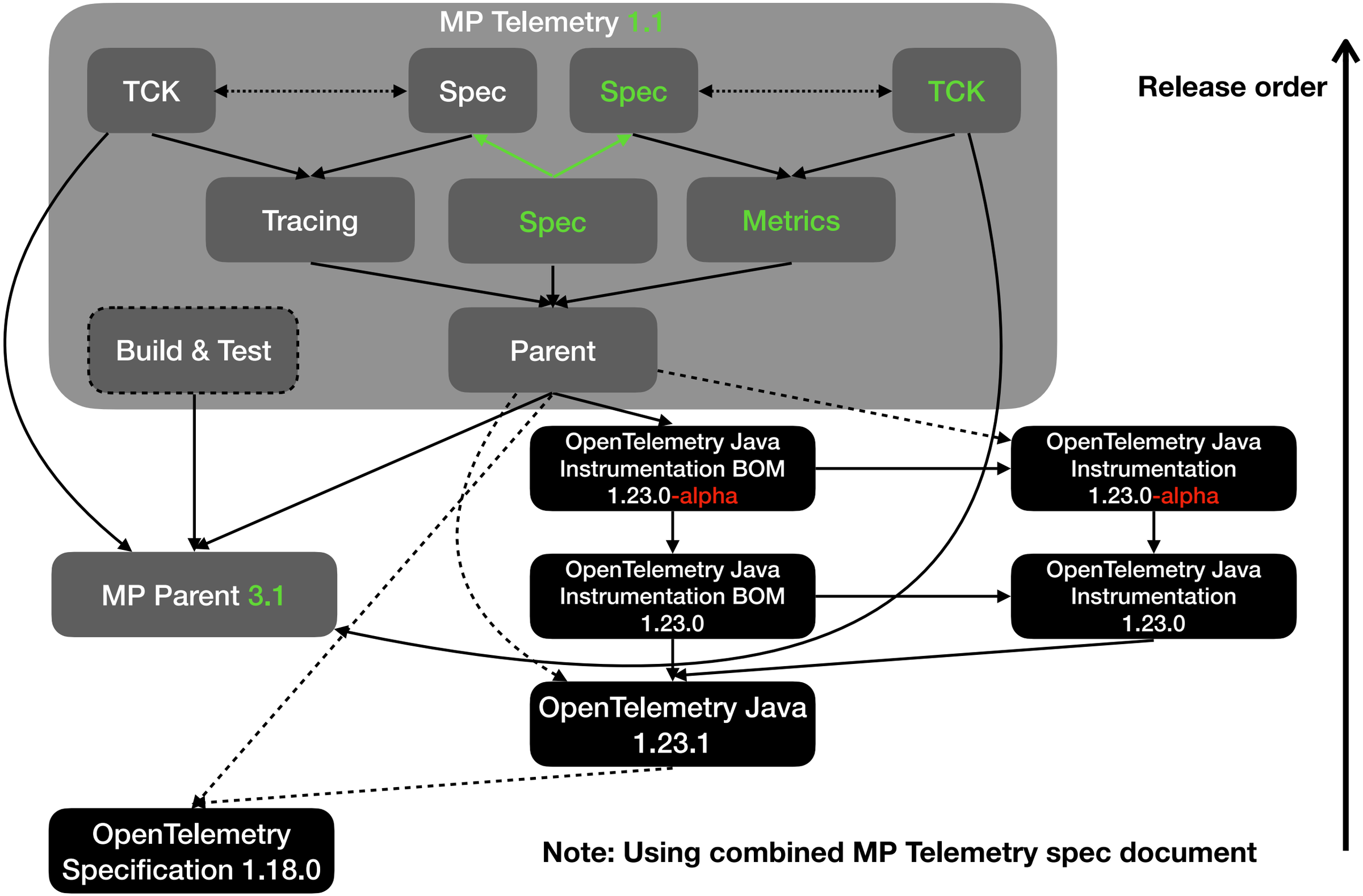
MP Telemetry & OpenTelemetry



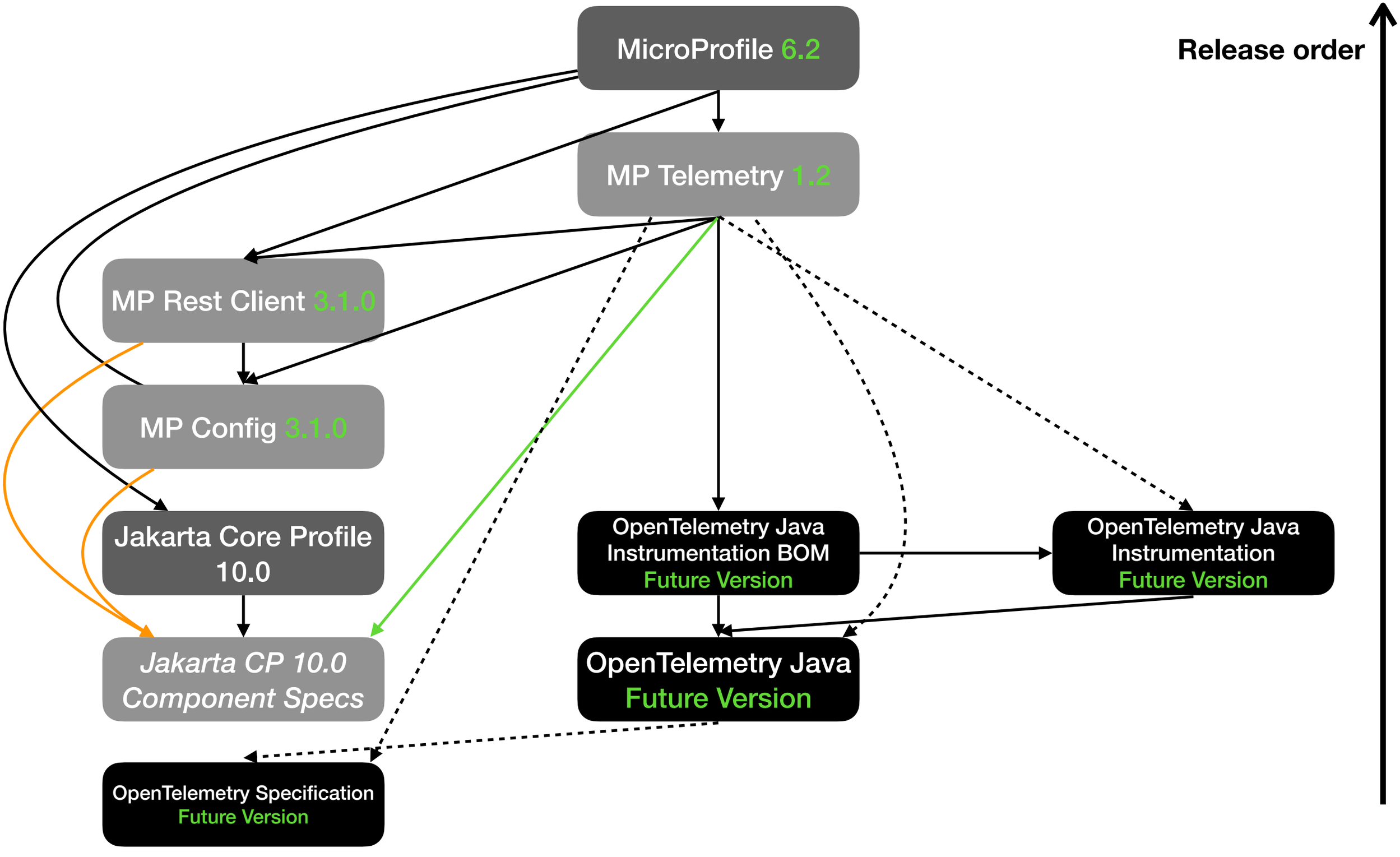
MP Telemetry & OpenTelemetry Detail



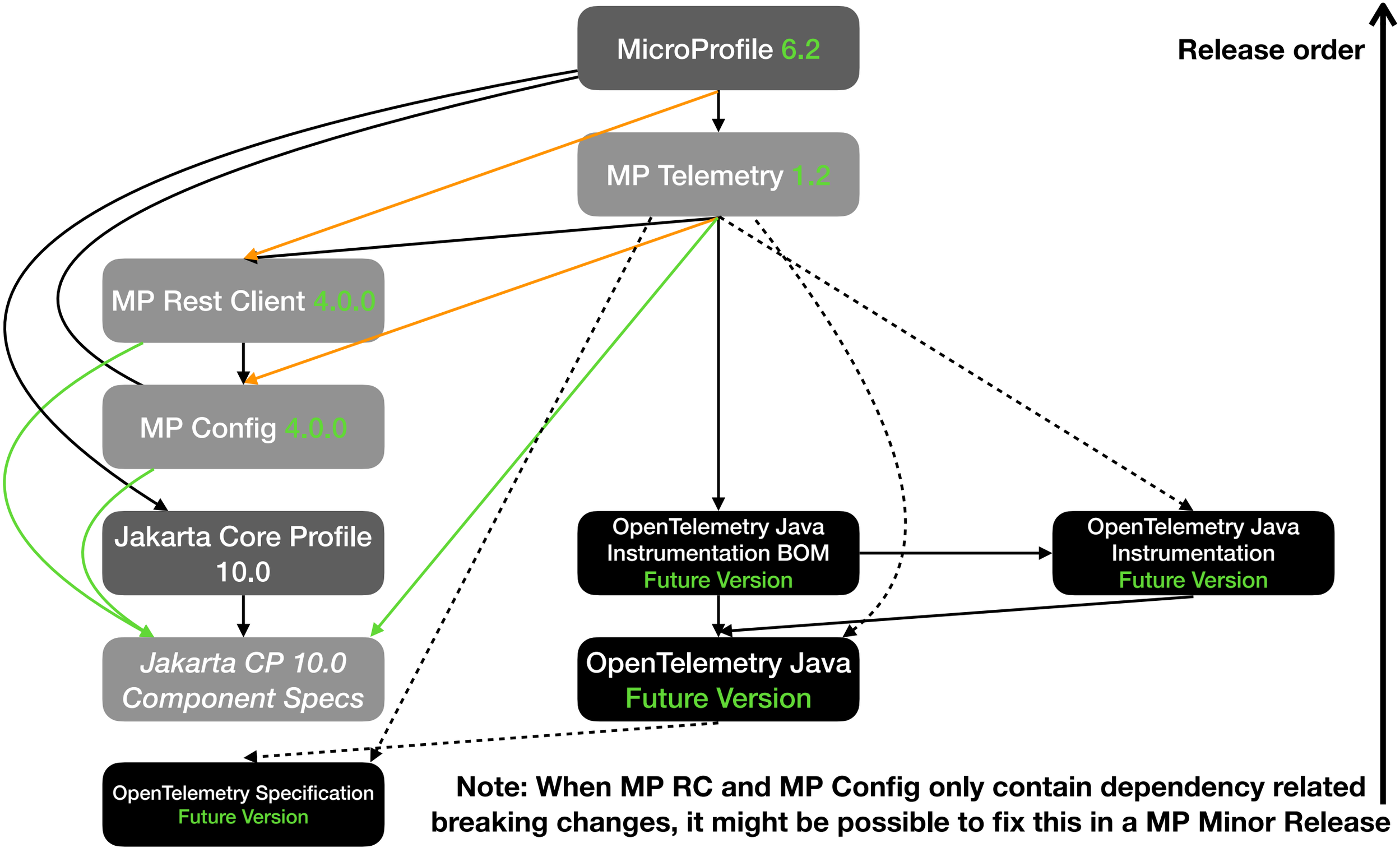
MP Telemetry & OpenTelemetry Detail



MP Telemetry & OpenTelemetry

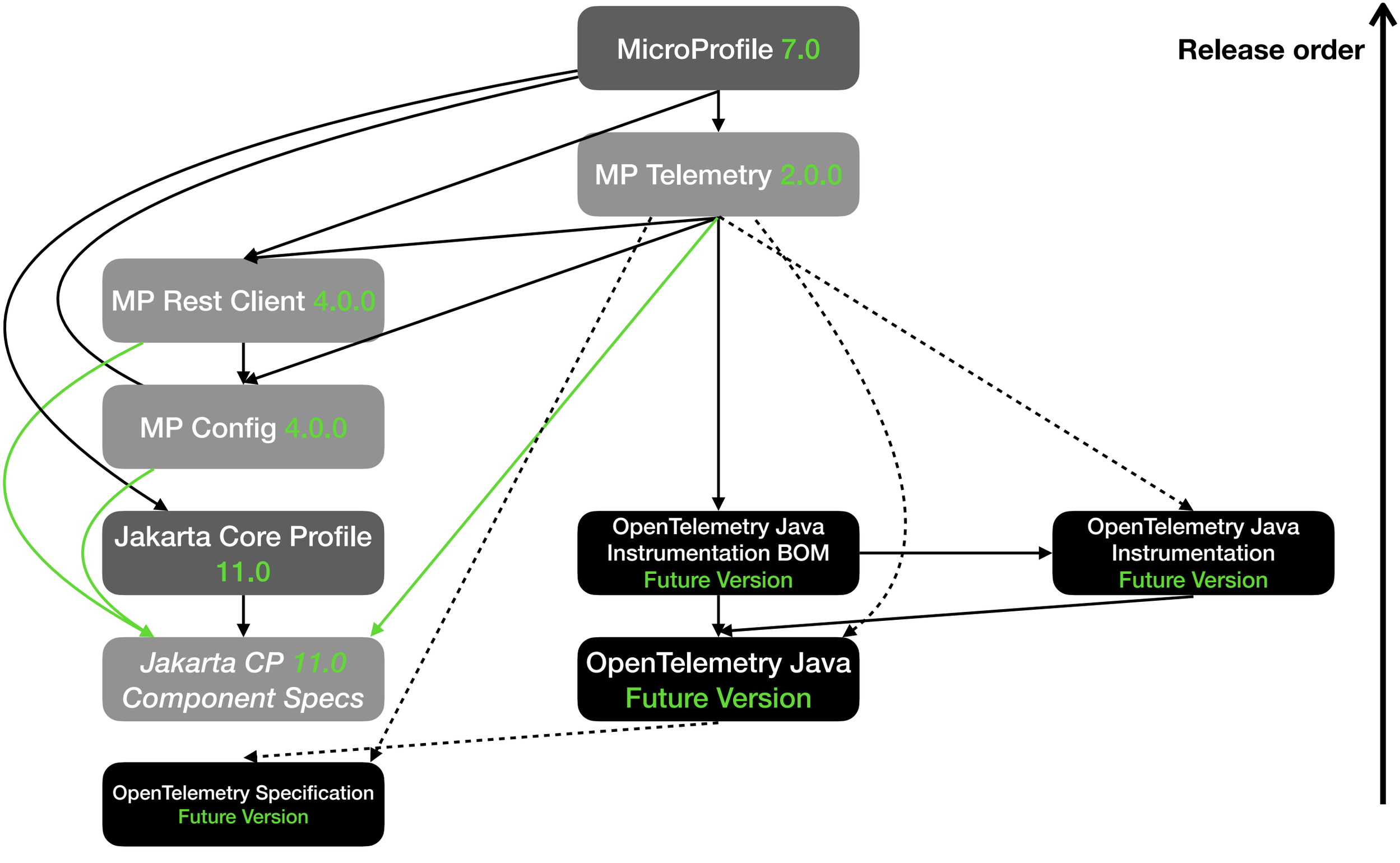


MP Telemetry & OpenTelemetry



Note: When MP RC and MP Config only contain dependency related breaking changes, it might be possible to fix this in a MP Minor Release

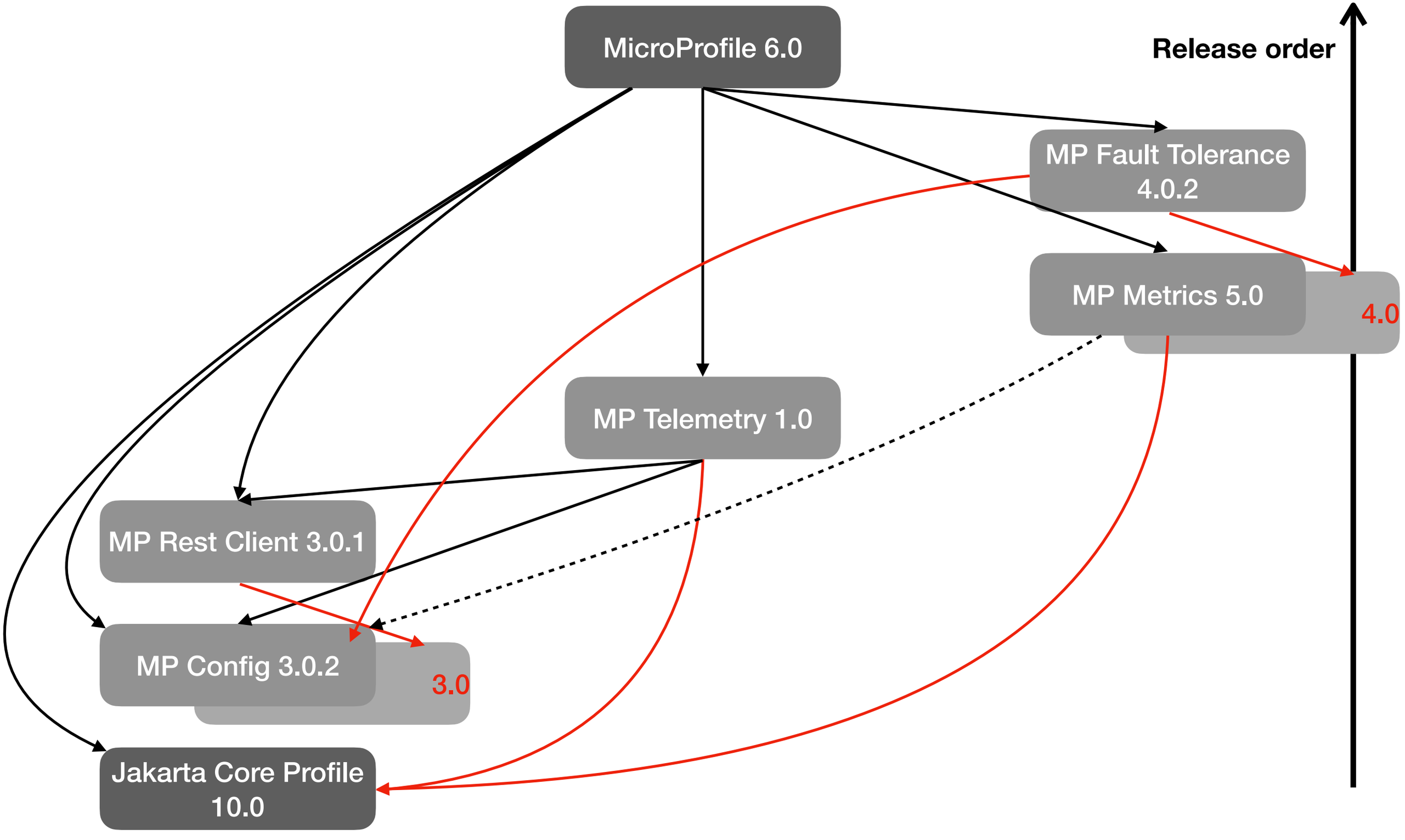
MP Telemetry & OpenTelemetry



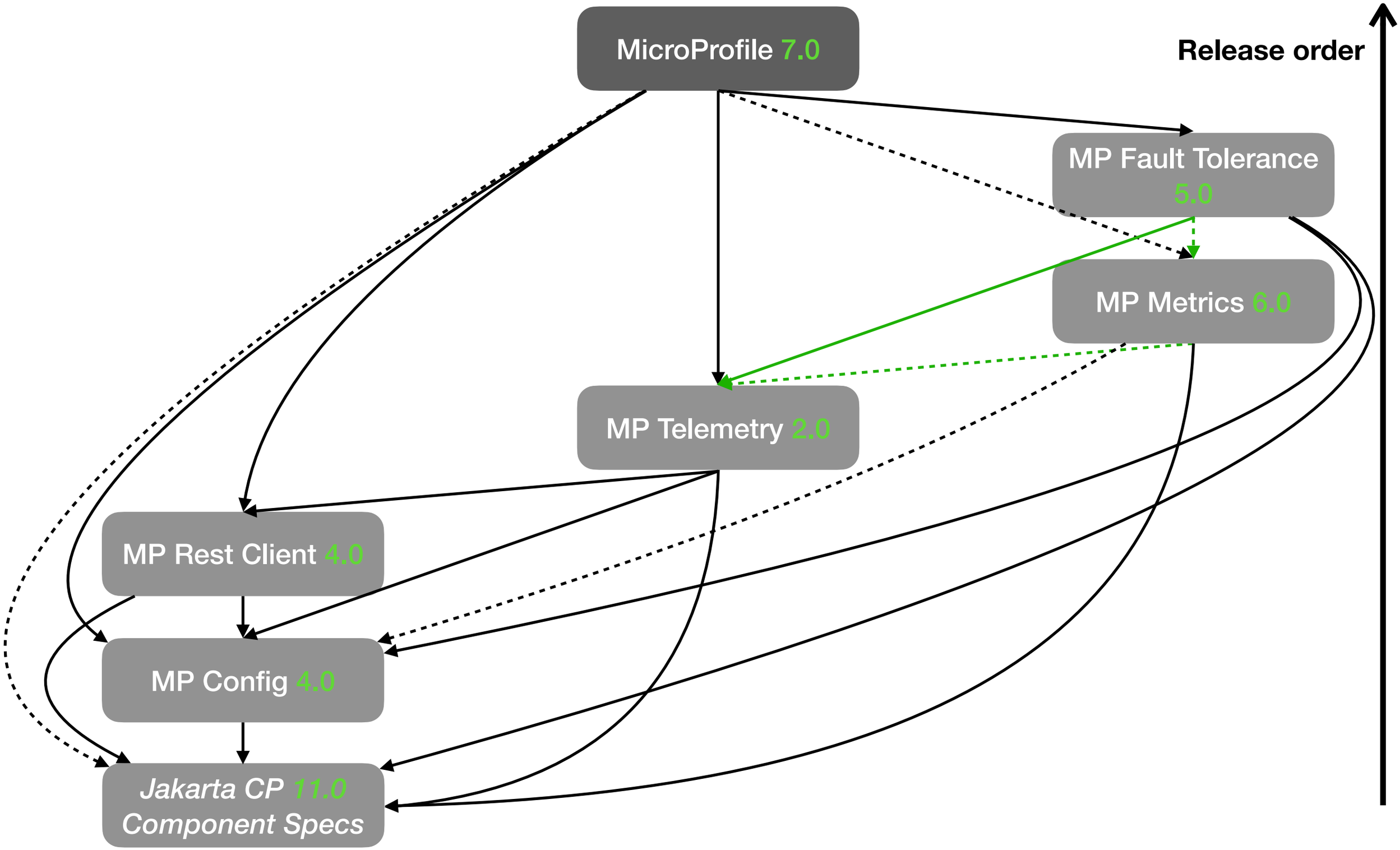
MP Metrics & MP Telemetry

- Issue
  - MP Metrics is not supported by all vendors anymore
  - MP Metrics may become replaced by MP Telemetry Metrics in future MP (Platform/umbrella) releases and will be a stand-alone spec then
  - OpenTelemetry Metrics is not fully stable yet (especially regarding Telemetry Data)
  - MP Fault Tolerance has a dependency to MP Metrics
- Solution
  - Align MP Metrics to MP Telemetry so they can coexist in implementations
  - Define configuration options and defaults, i.e. to prevent doublet metrics
  - Help stabilizing OpenTelemetry Metrics
  - Modify MP Fault Tolerance to depend on MP Telemetry (and optionally/alternatively on MP Metrics)

MP Metrics & MP Telemetry



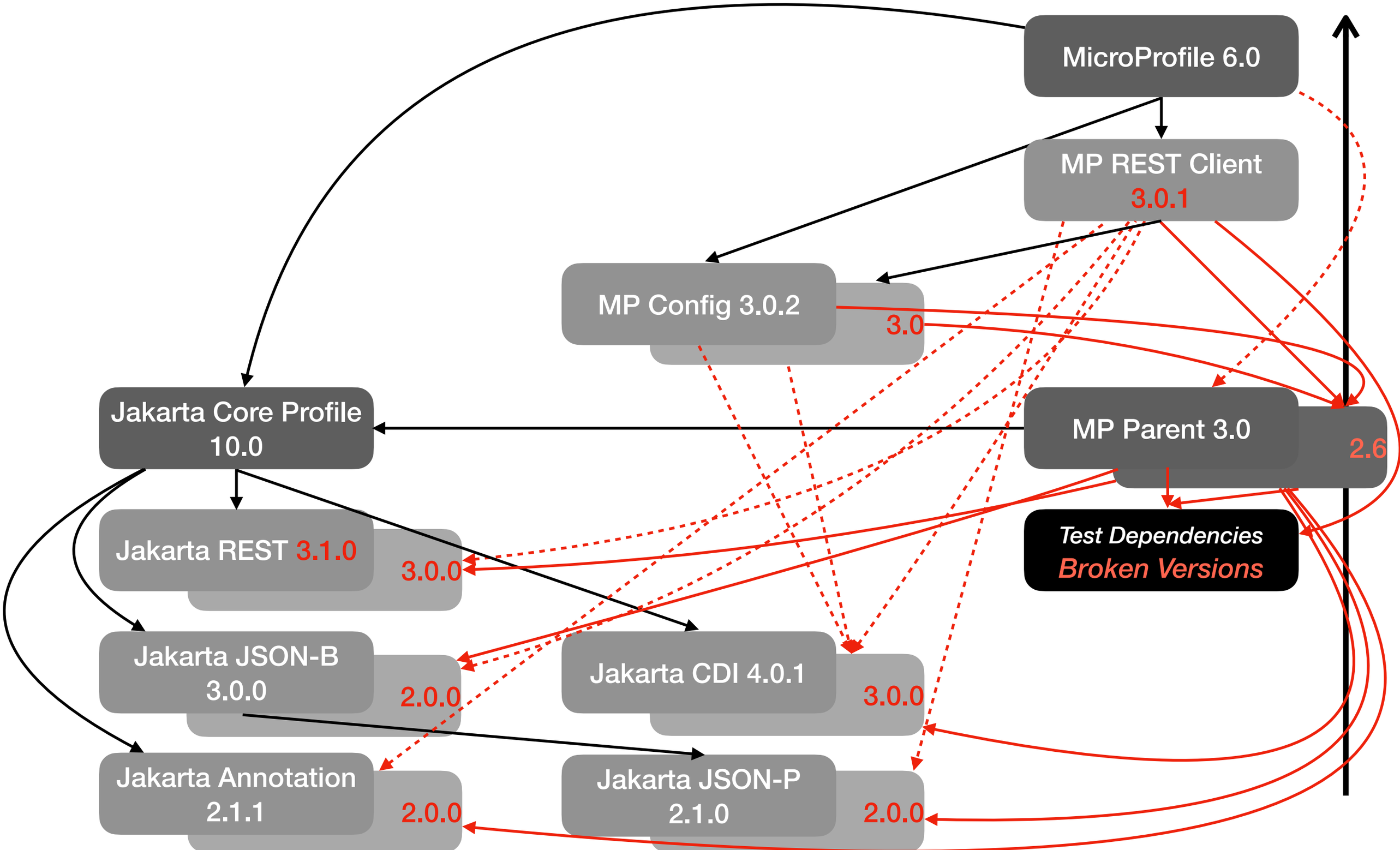
MP Metrics & MP Telemetry



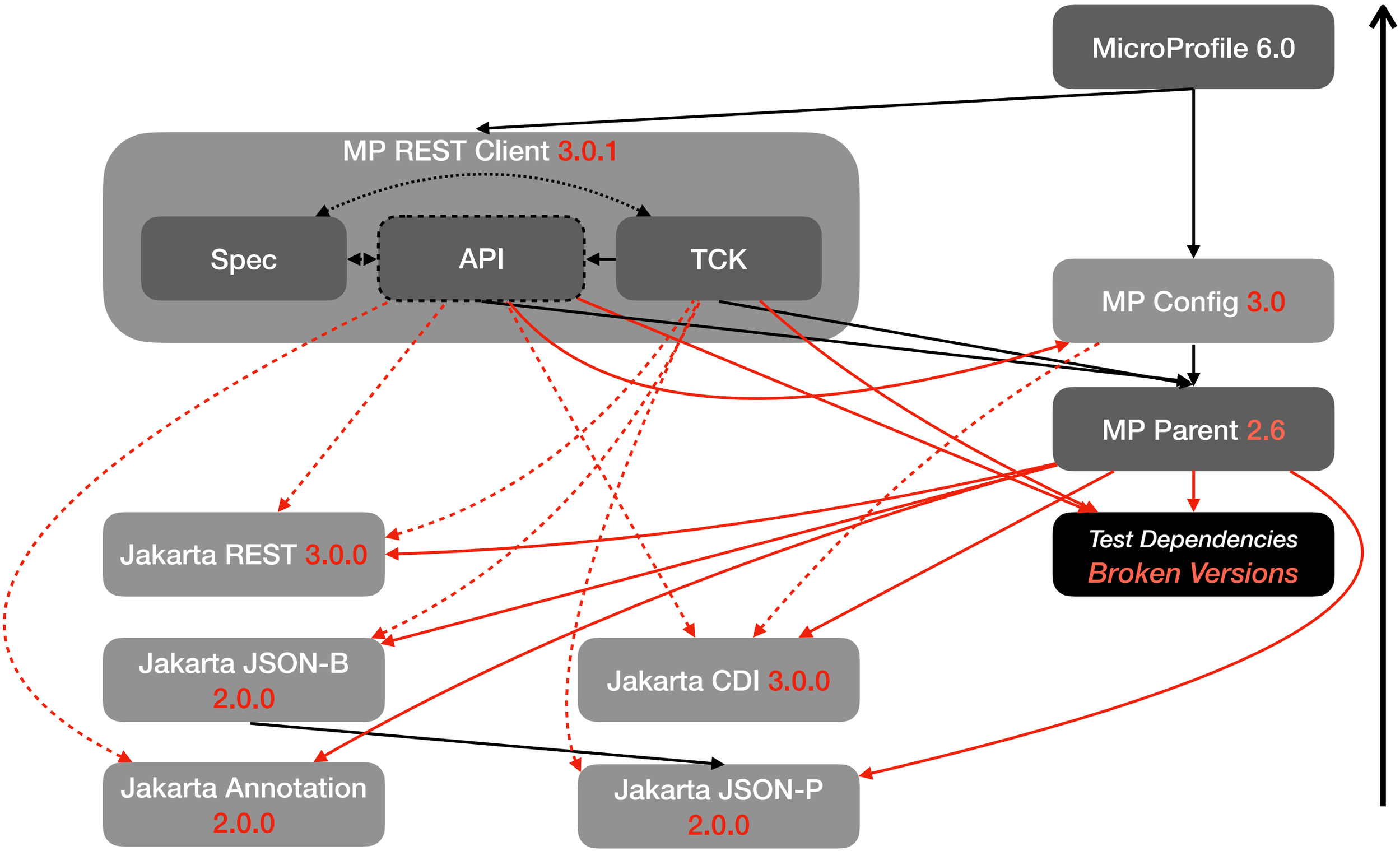
MP REST Client & Jakarta REST

- Issue
  - Dependencies to Jakarta EE 9.1 component specs (including Jakarta REST 3.0.0) - implementations got tested against different version (sometimes with deviating Major Releases like in CDI and JSON-B) than umbrella spec
  - Dependency to MP Config 3.0 instead of 3.0.2
  - TCK dependency to MP JSON-P
    - TCK dependency to outdated implementation (Geronimo Atinject 1.0 Spec & Geronimo Annotation 1.2 Spec, the last optional and for JavaDoc only) - this results in an indirect circular dependency!
    - Security issues (see separate topic)
- Solution
  - Update to Jakarta EE 10 Core Profile dependencies
  - Verify potential removal of TCK dependencies
  - Fix security issues
  - Update dependencies

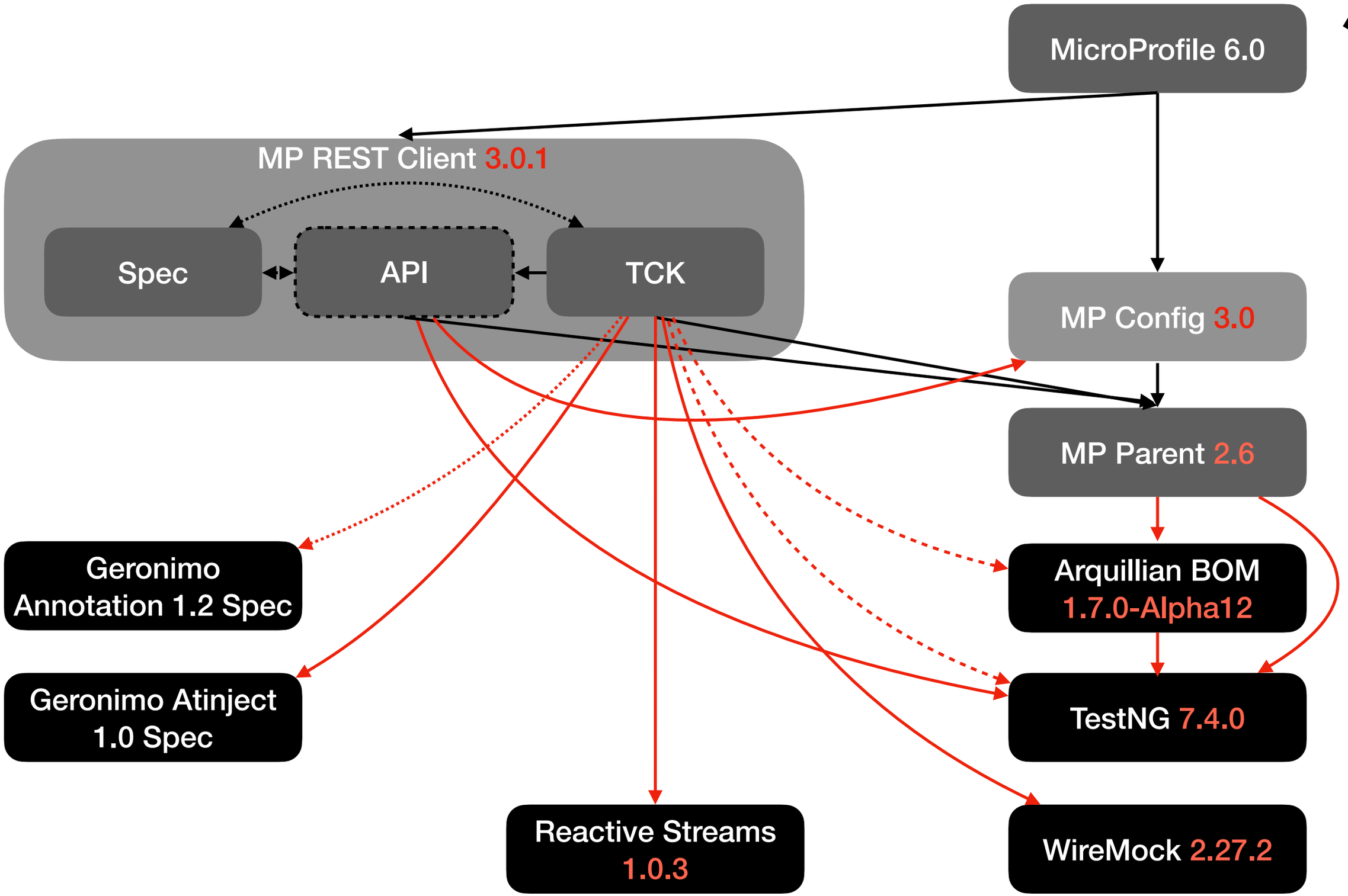
MP REST Client & Jakarta REST



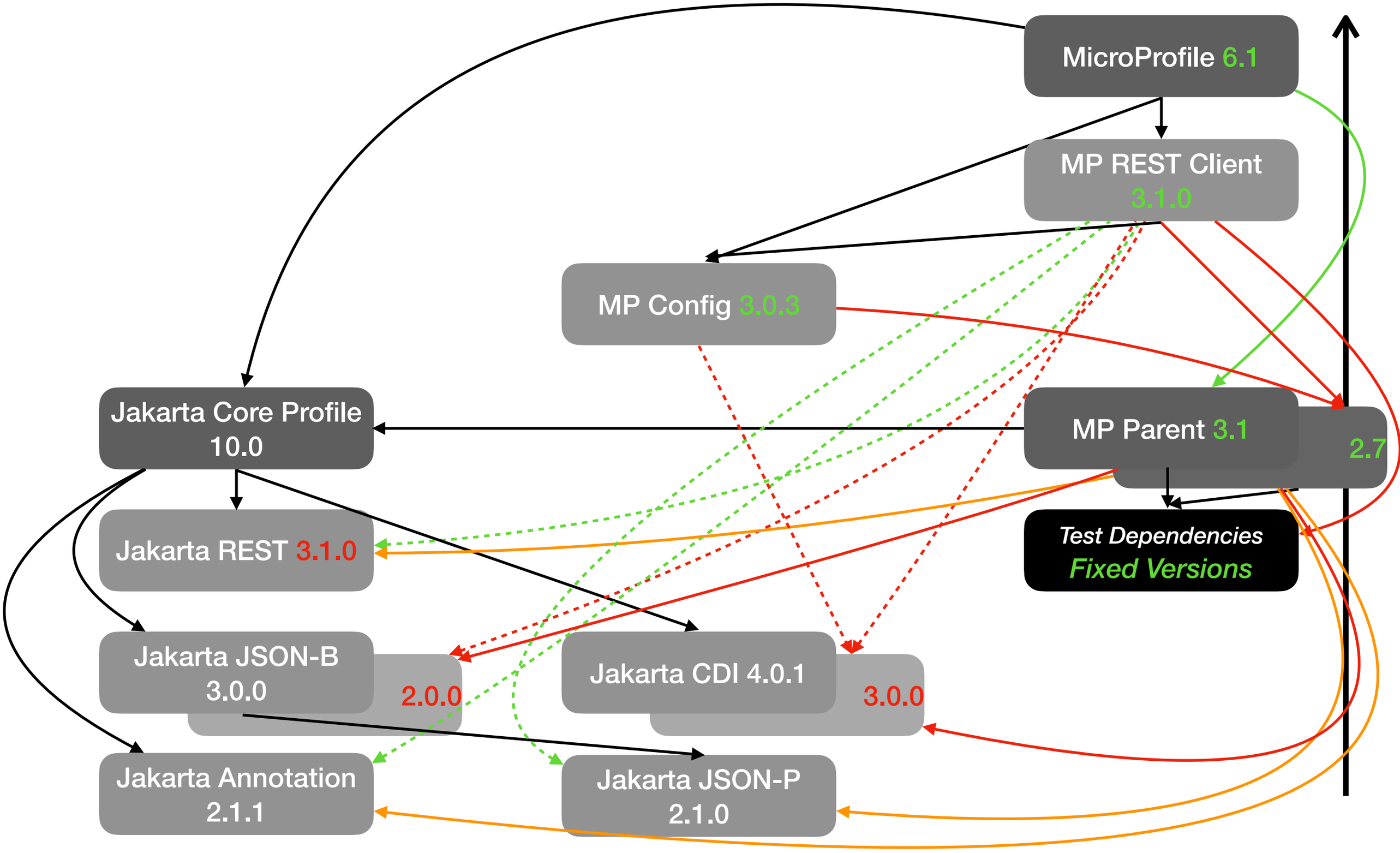
MP REST Client & Jakarta REST



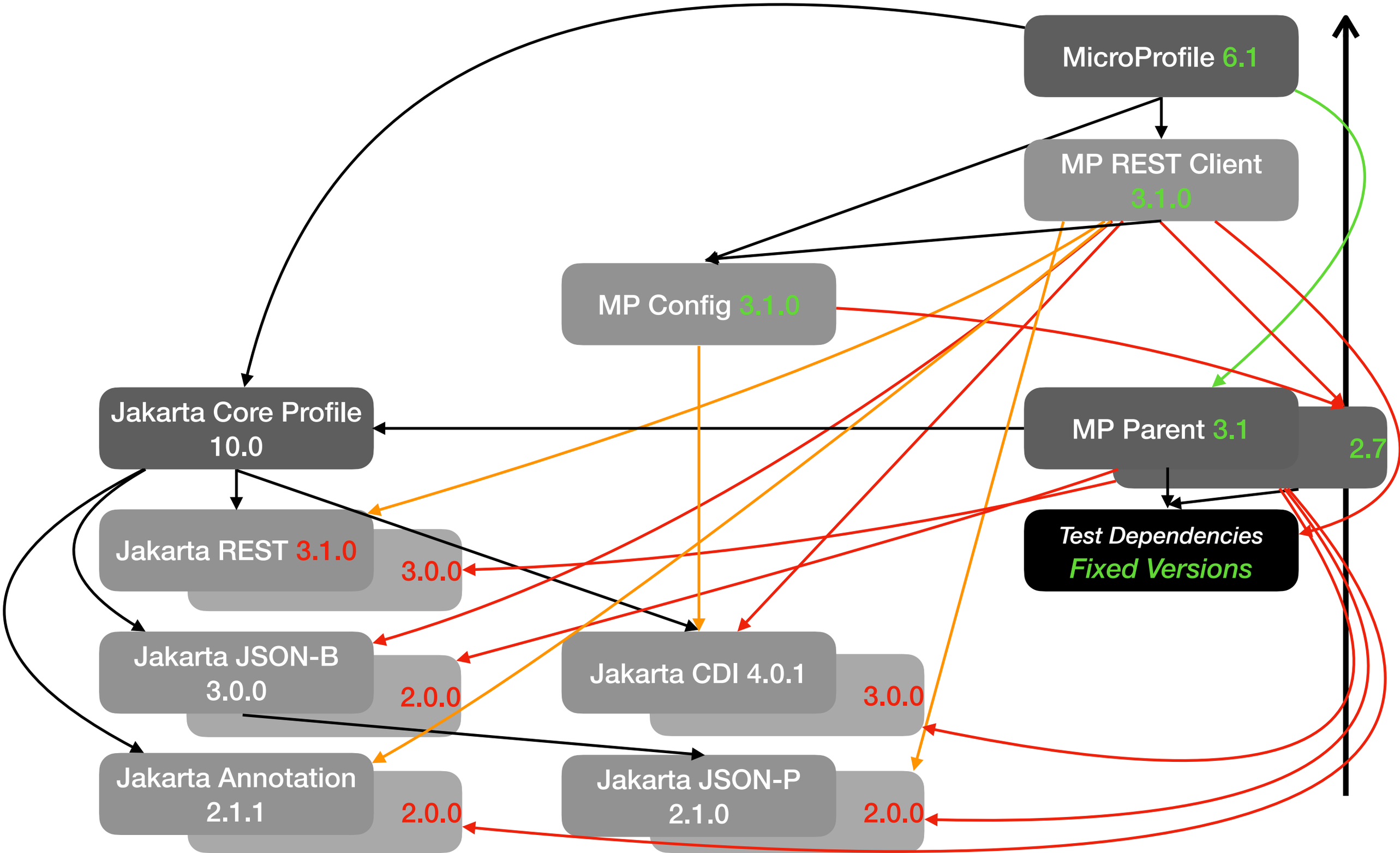
MP REST Client & Jakarta REST



MP REST Client & Jakarta REST



MP REST Client & Jakarta REST



MP REST Client & Jakarta REST