12 Verteilungsdiagramm

Inhalt

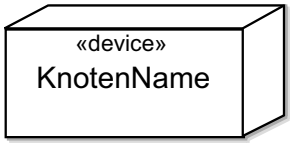
12.1 Allgemein

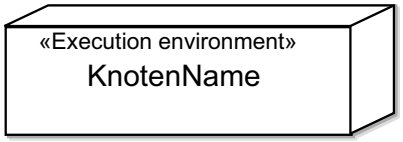
- **Verteilungsdiagramm =** engl. **Deployment Diagram** (**Deployments**)
  (in deutschsprachiger Literatur auch **Einsatzdiagramm** genannt)
- **Verteilungsdiagramm** zeigt die eingesetzte
  - *Hard- und Softwaretopologie* und
  - das zugeordnete *Laufzeitsystem*
- *Hard- und Softwaretopologie* umfasst Verarbeitungseinheiten in Form sogenannter
  - **Knoten** sowie
  - **Kommunikationsbeziehungen**

  zwischen diesen Verarbeitungseinheiten
- *Laufzeitsystem* beinhaltet ausschließlich implementierte UML-Modellelemente in Form so genannter **Artefakte**.
- Vollständige **Spezifikation der Konzepte siehe [OMG 2015]**; → Literaturangaben!)
- Hinweis: unterschiedliche Unterstützung durch Tools

12.2 Knoten

- zwei Arten von Knoten
  - Geräte (**<<devices>>**)
    i.d.R. Hardware mit Rechnerkapazität,
  - *Laufzeitsystem* wird auf Geräte 'verteilt'
    Ausführungsumgebungen
    (**<<executionEnvironment>>**)
    Software (z.B. OS, Browser)

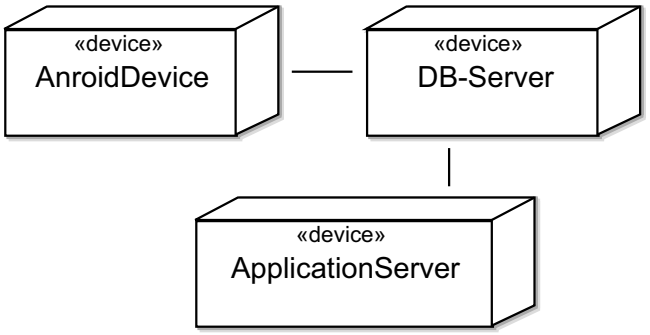




- **Notation**: Quader (3D) mit Stereotyp und Namen
- Knoten können geschachtelt werden
- Knoten können interne Struktur aufweisen: Rollen und Ports, auf die Artefakte verteilt werden können.
- Besondere Eigenschaften von Knoten, wie z.B. ihre Kapazität und Ausfallsicherheit, sind in UML nicht vordefiniert, können jedoch durch benutzerdefinierte Stereotype spezifiziert werden.

12.3 Kommunikationsbeziehungen zwischen Knoten

- Knoten können durch Kommunikationsbeziehungen miteinander verbunden werden
- **Notation**: Kommunikationsbeziehung wird — als Spezialform einer **Assoziation** — als durchgezogeneLinie dargestellt, die gerichtet oder ungerichtet sein kann



- spezifiziert Austausch von Nachrichten und Signalen
- Analog zu Knoten kann auch diese durch benutzerdefinierte Stereotype präzisiert werden, wie z.B. «internet» für eine Kommunikationsbeziehung auf Basis der Internet-Infrastruktur, oder z.B. «ethernet» zur Charakterisierung eines lokalen Netzwerks

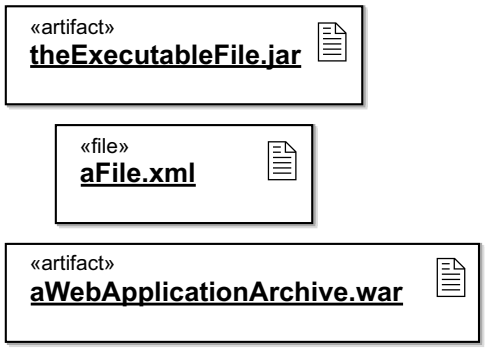
12.4 Artifakte

- Artefakte sind jene Modellelemente, die für die Ausführung auf die verschiedenen Knoten verteilt werden können

- In UML 2.x können nur Artifakte verteilt werden

- Artefakt
  - ist inhärent mit der Implementierung eines Systems verbunden
  - repräsentiert eine physische Informationseinheit, die im Rahmen eines Softwareentwicklungsprozesses oder einer Systemausführung erstellt oder verwendet wird
  - kann ein Modell sein
  - kann eine Beschreibung sein
  - kann eine Software darstellen
  - kann gespeichert und manipuliert werden
  - kann Attribute und Operationen aufweisen
  - kann Beziehungen zu anderen Artefakten eingehen
  - kann geschachtelt sein (Aggregat von Artifakten)
  - kann instanziiert werden

- **Notation** von Artifakten
  - Rechteck mit Schlüsselwort **«artifact»** und einem optionalen Dokumentsymbol
  - Artefaktinstanzen: Name unterstrichen



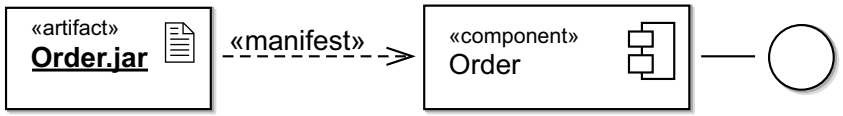
- UML Standard Stereotypen für Artefakte
  alle Stereotype eine Spezialform des Stereotyps «file»:
  - **«file»** – eine physische Datei im Kontext des zu entwickelnden Systems
  - **«document»** – eine generische Datei, die weder eine «source»-Datei noch eine «executable»-Datei darstellt
  - **«executable»** – eine Programmdatei, die von einer Verarbeitungseinheit ausgeführt werden kann
  - **«library»** – eine statische oder dynamische Bibliotheksdatei
  - **«script»** – eine Skriptdatei, die von einer Verarbeitungseinheit interpretiert werden kann
  - **«source»** – eine Datei, die in eine «executable»-Datei kompiliert werden kann

- spezielle Artefakte für verschiedene Implementierungsumgebungen,
  z.B. «jar» als spezieller «executable»-Stereotyp für EJB,
  werden über entsprechende Profile definiert

12.5 Manifestation

- Beziehung zwischen logischen **Modellelementen** und **physischen Artefakten** wird als **Manifestation** bezeichnet

- **Notation**: (spezielle Form der Abstraktionsabhängigkeit)
  strichlierter Abhängigkeitspfeil mit dem Schlüsselwort **«manifest»**



- ein Modellelement kann durch mehrere Artefakte manifestiert werden

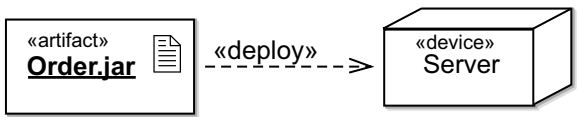
- ein Artefakt kann mehrere Modellelemente manifestieren

- Beispielsweise können die Artefakte *Sourcedatei*, ausführbare *Programmdatei* und *Bibliotheksdatei* eine Klasse, die im Modellierungsprozess entstanden ist, manifestieren.

12.6 Verteilung von Artifakten auf Knoten

- Beschreibt das Zusammenspiel
  - des *Laufzeitsystems* bestehend aus Artifakten und
  - der *Knoten* der Hard- und Softwaretopologie

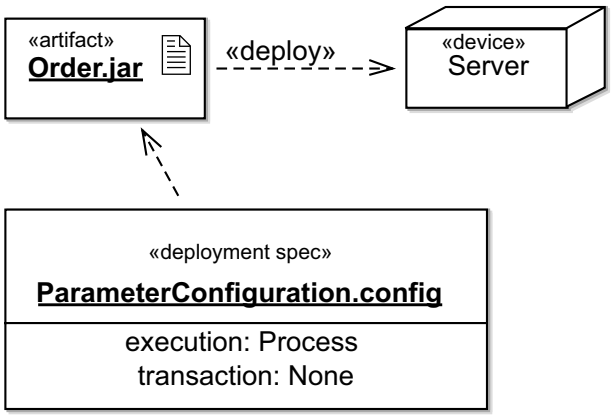
- **Notation**: Die Verteilung eines **Artefakts** auf einen **Knoten** wird durch einen strichlierten Abhängigkeitspfeil mit dem Schlüsselwort **«deploy»** notiert.



- Die Verteilungsbeziehung kann auf Typ- oder Instanzebene modelliert werden.

- Alternativ: grafische Schachtelung, indem die Artefaktsymbole direkt innerhalb des Knotensymbols angegeben werden,
  oder indem einfach die Namen der Artefakte in das Knotensymbol eingetragen werden.

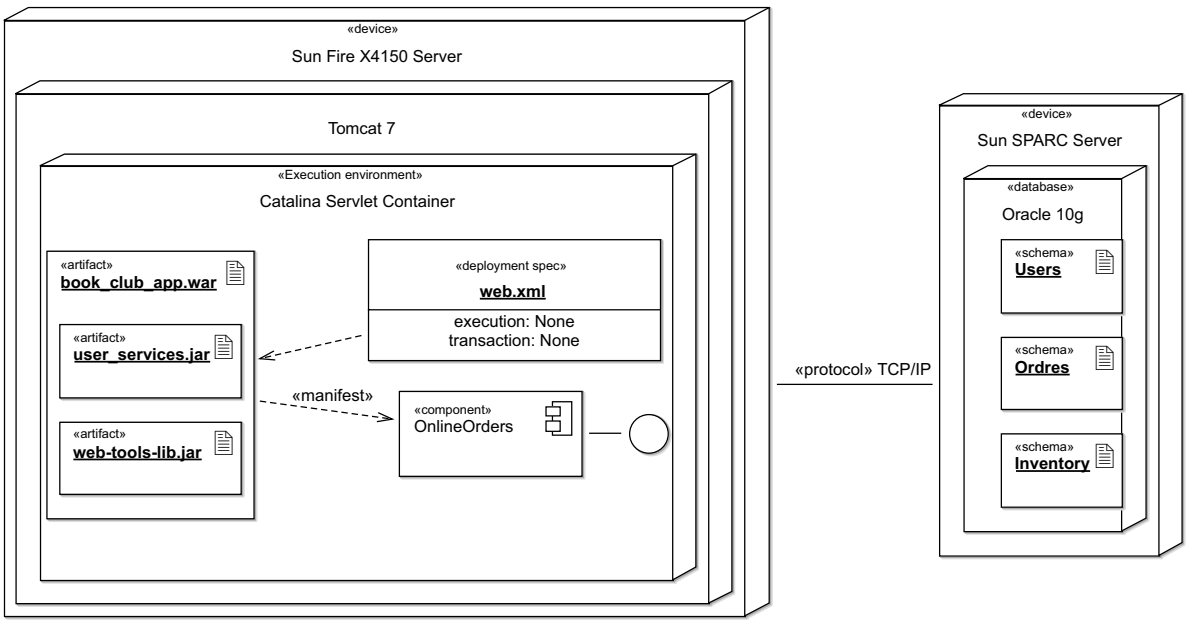
- Details zur Verteilung eines Artefakts auf Knoten, wie beispielsweise Installationspfade, sowie Details zur Abarbeitung eines Artefakts, z.B. die Priorität der Abarbeitung und die maximale Dauer, können durch entsprechende Parameter in einer **Verteilungsspezifikation** festgelegt werden



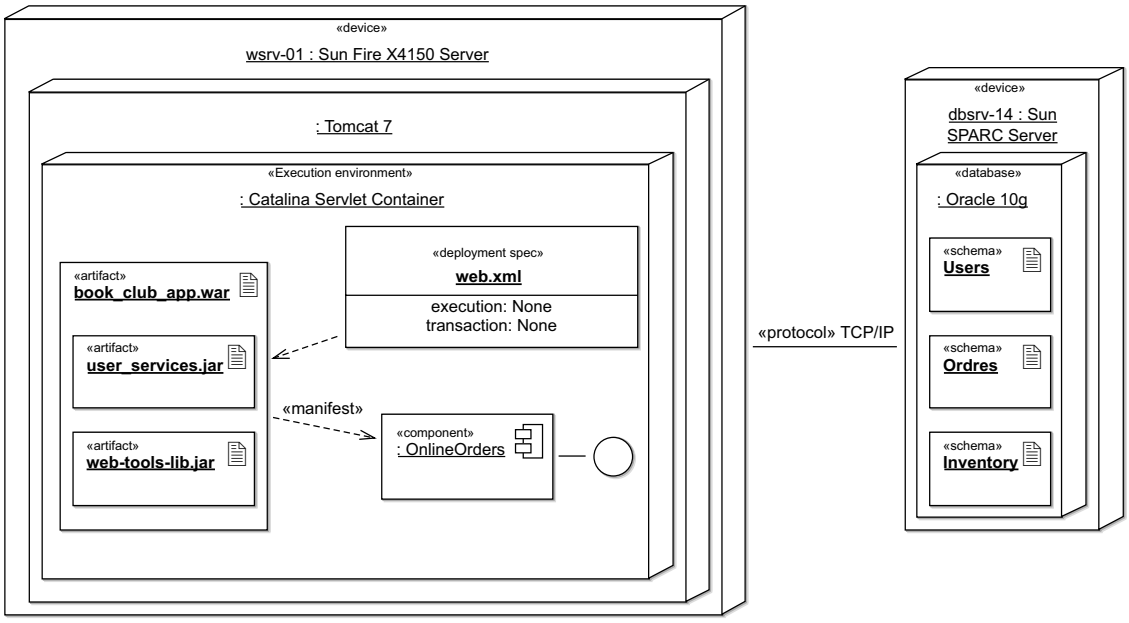
12.7 Verteilungsdiagramm auf Typ- und Instanzebene

- Verteilungsdiagramme können auf zwei Ebenen spezifiziert werden, lt. UML-Spezifikation
  - **Typebene**
    - Verknüpfung von Arten von Knoten mit Arten von Artifakten (kinds of DeploymentTargets mit kinds of DeployedArtifacts)
    - Beispiel: “application server” mit einem “order entry request handler”
  - **Instanzebene**
    - Verknüpfung von konkreten Knoten mit konkreten Artifakten (particular DeploymentTargets instances mit particular DeployedArtifacts instances)
    - Beispiel: zwei konkrete application server (“app-server1”, “app-server-2”) als Knoten für sechs verschiedene, konkrete “request handler”

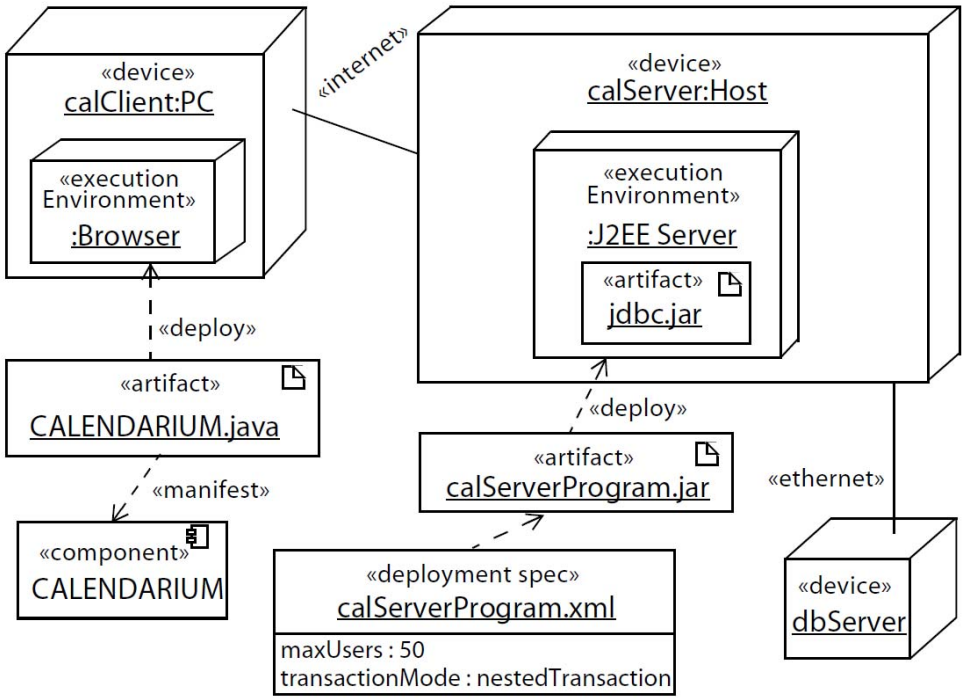
❒ Beispiel auf Typebene (Ausführung Toolabhängig):



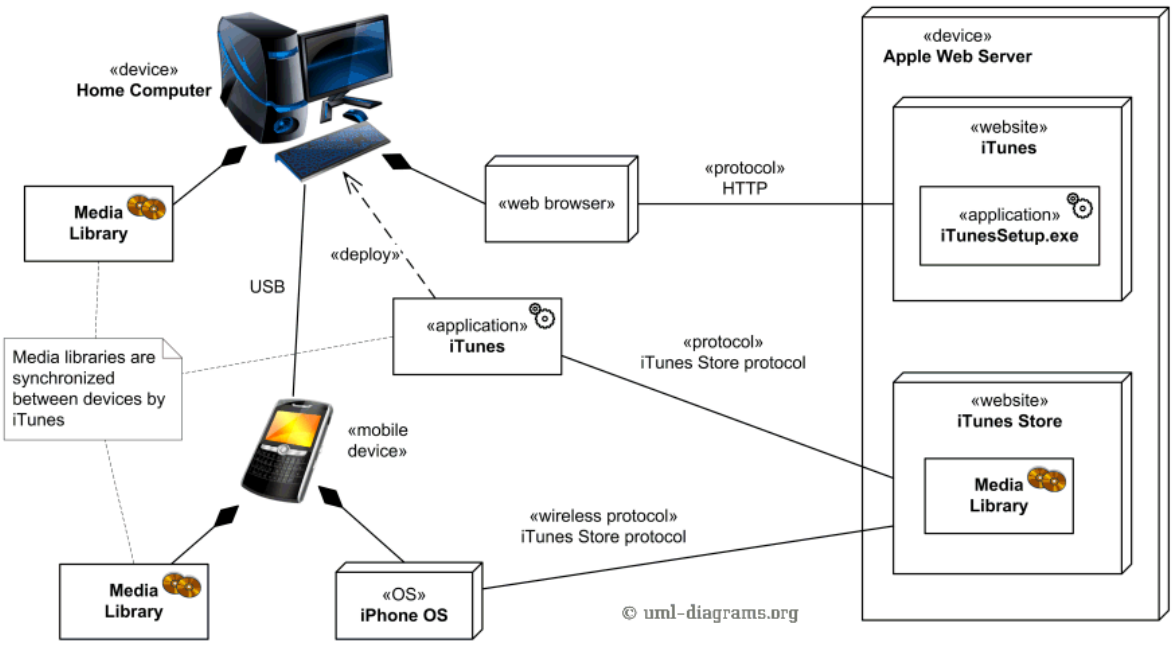
❒ Beispiel auf Instanzebene (Ausführung Toolabhängig):



❒ Beispiel (Erläuterungen siehe [UML@Work])



❒ Weitere Beispiele
Siehe: http://www.uml-diagrams.org/deployment-diagrams-examples.html



Literatur und Quellen:

- [OMG 2015] OMG Unified Modeling Language™ (OMG UML), Version 2.5
  Normative Reference: http://www.omg.org/spec/UML/2.5
  OMG Document Number formal/2015-03-01

- [UML@Work]
  Kapitel 3.6 in:
  Martin Hitz, Gerti Kappel, Elisabeth Kapsammer, Werner Retschitzegger:
  UML @ Work - Objektorientierte Modellierung mit UML2.
  dpunkt Verlag 2005 / 3. aktualis. u. überarb. Aufl. 2005.
  ISBN-13: 9783898642613 ISBN-10: 3898642615

- Tool zur Vertiefung: BEE-UP Modelling Tool:
  http://austria.omilab.org/psm/content/bee-up/info
  http://www.omilab.org/web/guest/omilab-in-education/cmmc