Technisch:

* Schichtenarchitektur
  + Beschreibung / Skizze
  + Trennung
  + Erklärung der Funktion der einzelnen Schichten
  + Abstraktion zur DB-Schicht
  + Domäne
    - Domänenlogik
  + Controller
    - Use-case bezogen (Domäne)
    - Transaktionen
* Validierung
  + Gui
* Exceptions
  + Weitergabe bzw. neue pro Ebene
* Interfaces (Subpackages)
* Mapping
  + Namenskonventionen
  + Dynamische Erzeugung von Objekten
  + Austausch der Persistenz-Schicht relativ einfach (bzw. ohne Änderung der Domänenschicht möglich)
* Controller
  + State-Pattern
* Patterns allgemein
  + Eventuell zu den einzelnen Schichten/Controller hinzufügen
  + Observer
  + Mapper
  + State
  + Proxy
* „Zugriffsschutz“(Information-Hiding): Interfaces, Schichtentrennung
* Transaktionen
  + Steuerung (wer, wie)
  + Verwendung
* Hibernate-Framework
  + Abfragen im …-Stil (nicht native-SQL), weil
    - Abstrakter
    - Optimierter
    - Änderungen ersichtlich
    - Teilweise Überprüfung zur compile-zeit
  + Abstraktion zur DB
    - Austausch der DB relativ leicht möglich (!kein native-SQL)
* SWING-Framework
  + War Vorgabe (wegen späterer Integration von SW anderer Teams)
* Packagestruktur
  + Begründung
    - Übersichtlichkeit
      * Logisch
      * Schichten
      * Information-Hiding

Vollbrecht:

- Designdokumentation:

- Packagediagramm zur Übersicht der Schichtenarchitektur, Packagediagramme einzelner Schichten

- Klassendiagramme

- Sequenzdiagramme, mindestens einen vollen Durchlauf durch alle Schichten.

- Textdokument mit der Beschreibung und Begründung wichtiger Designentscheidungen