1 Virtuelle Produktentwicklung

1.1 CAx - Methoden

1. Semi empirisch-/physikalische Modelle/Simulation
   P.1 F.69
   - empirisch: keine mathematische Beschreibung, Annäherungen durch Messung
   - physikalisch: physikalische Regeln und Axiome, mathematische Beschreibung eines Prozesses, numerische Lösung, Vorteil: physikalische Bedeutung
   - semi-empritisch: empirisch+physikalisches Modell zusammen
2. Kirchhoff'sche Einteilung der Modellierung
   P.1 F.71

System	vernachlässigte Kräfte	angewandte Kräfte
unbewegt	Geometrie CAD	(Elasto-)Statik FEM, BEM
bewegt	Kinematik CAD, MBS	Dynamik MBS, FEM

3. Welche Arten von Diskretisierung?
   P.1 F.72
   - Diskretisierung von Dichte, Steifheit, Viskosität
   - Dimensionen: unendlich $\rightarrow$ endlich
4. Welche Unterschiede zwischen den Typen

1.2 Cax - Workflows

5. Workflows beschreiben, Wie/Was läuft ab. P.1 F.88
   - DMU (Digital Mock Up)
     - ditigale Dummies die eine vereinfachte Darstellung des Produktes beinhalten
     - inkludierte Information
       * Produkt Geometrie (Volumen und/oder Oberflächen)
       * Produktstruktur
     - dienen Kollisionsüberprüfung, Simulation von Zusammenbau und Montage, ...
     - Als Grundlage für DMUs dienen vereinfachte 3D-CAD Daten des virtuellen Produkt Modells.
     - Durch die Vereinfachung ist die Genauigkeit des Modells reduziert.
     - Im DMU Prozess ist eine direkte Änderung der Geometrie nicht möglich.
   - FEM (Finite Elemente Methode)
     - Genutzt für Belastungs-/, Deformations-/, Schwiungs-/, Themodynamikberechnungen
     - FEM Berechnungen basieren auf einer angenäherten Geometrie, abgeleited von einem 3D-CAD Modell. Abhängig vom Ziel der Simulation werden verschiedene Annäherungen vorgenommen.
     - Der Datentransfer vom 3D-CAD Modell uns FE-Programm erfolgt mit einem Diskretisierungsprozess
     - Randbedigungen der Lasten (Kräfte, Momente, ...), Einschränkungen (fixe Komponenten, Lager, ...), Materialeigenschaften, Temperatur, etc. werden im FEM-Programm definiert.
     - Nach der Simulation wird das Resultat ausgewertet. Änderungen werden an der master-Geometrie im 3D-CAD Programm und nicht in der FEM-Software vorgenommen.
   - CFD (Computational Fluid Dynamics)
     - CFD-Simulationen ermöglichen die Berechnung und Optimierung von Gas- und Flüssigkeitsströmungen.
     - Ähnlichen den FEM-Berechnungen wird die Geometrie mit einem mesh angenähert.
     - Datentransfer von 3D-CAD Modell zu CFD-Programm wird mit neutralen, standardisierten Datenformaten (STEP, IGES, ...) vorgenommen

  - Randbedingungen werden direkt im CFD-Programm festgelegt.
  - Nach der Simulation wird das Resultat ausgewertet. Änderungen werden an der „Master-“ Geometrie im 3D-CAD Programm und nicht in der CFD-Software vorgenommen.
- MBS (Mehrkörpersimulation)
  - Genutzt für kinematische Berechnungen und Optimierung des zusammengebauten und beweglichen Produkts.
  - Das MBS-Modell basiert auf der CAD-Produkt-Struktur, wobei Körper, Verbindungen und Gelenke als Starrkörper angesehen werden.
  - Randbedingungen (Kräfte, Momente, Massen, Freiheitsgrade der Bewegung) werden im MBS-Programm definiert.
  - MBS Modelle setzen sich aus vereinfachten geometrischen Elementen zusammen welche die relevanten Daten für die kinematische Berechnung beinhalten.
  - Nach der Simulation wird das Resultat ausgewertet. Änderungen werden an der „Master-“ Geometrie im 3D-CAD Programm und nicht in der MBS-Software vorgenommen.

6. Prozessworkflow CAD-VR P.1 F.101

   - Virtual Reality beschreibt eine computergenerierte Umgebung welche als Benutzerinterface fungiert.
   - VR basierte Studien inkludieren den Nutzer in die virtuelle Umgebung
   - Echtzeit Interaktionen der Geometrie / Funktionalität unterstützen eine Beurteilung des Modells
   - Die Vorstellung des der Manipulierbarkeit ermöglicht eine lebensnahe Erfahrung des virtuellen Produkts.
   - Das VRML-Datenformat (Virtual REality Modeling Language, 1994) wurde entwickelt um 3D-Modelle darzustellen und Nutzerbasierte Interaktionen zu integrieren.
   - VRML-Daten, welche vom 3D-CAD master-Modell abgeleitet werden inkludieren vereinfachte Geometrien ausschließlich in aktueller Ausführung, ohne Verlaufsdaten.

1.3 Product Data Management

7. Was ist PDM?
   P.1 F.112 PDM (Product Data Management) beinhaltet alle Aufgaben einer Organisationseinheit für die Identifizierung, Bereitstellung, Archivierung von produktrelevanten Daten während der Produktentwicklung.

8. Warum PDM?
   P.1 F.113-115

   - Hilft Organisationen den Daten- und Informationsfluss während des Entwicklungsprozesses zu organisieren.
   - Die Verwaltung des Daten-, Prozess-, und Dokumentenstroms über den gesamten Lebenszyklus bilden die Grundlage für die virtuelle Produktentwicklung.
   - Komplexe Produktstrukturen oder Variationen erzeugen zahlreiche Parameter und Informationen. Ein leistungsstarkes PDM System unterstützt den Datenaustausch zwischen den verschiedenen Phasen der Entwicklung.

9. Concepts of the virtual product development
   P.1 F.116

10. Nennen Sie Daten die in einem PDM System verwaltet werden können

    - Geometriedaten
    - 2D-Zeichnungen
    - Produktstruktur
    - Ergebnisse von Analysen
    - Dokumente

11. Hauptfunktionen PDM
    P.1 F.119

- Produktstruktur-management
- Workflow-management
- Projekt-management
- Dokument-management
- Klassifikation / Varianten
- Beziehungen untereinander: Personen $\Leftrightarrow$ Prozesse $\Leftrightarrow$ Teile $\Leftrightarrow$ Dokumente $\Leftrightarrow$ Daten

2 Computer-Aided Design (CAD)

2.1 Geometrical representation models in CAD

12. Element Typen
    P.2 F.13-29

    - Drahtgitter: Punkte und Kurven definieren 2D- und 3D-Objekte
    - Flächen: Ebene und gewölbte Flächen im $\mathbb{R}^3$
    - Solid: Ermöglicht komplette geometrische Darstellung von realen Produkten in virtueller Umgebung
      - Schalengeometrie: geschlossene Schalten $\rightarrow$ Solid
      - Unterscheidung zwischen Innen und Außerhalb des Solids
      - Definition Materialeigenschaften (für Simulationen)
    - Hybride Modelle

13. Was ist ein Skelettmodell
    P.2 F.14 s.o.

14. Wie wird es angewandt?

15. Anwendung von Bool'schen Operationen bezogen auf Produktion
    P.2 F.25,27

    - Vereinigungs Operator (A$\cup$B)
    - Subtraktions Operator (A$-$B)
    - Schnittmenten Operator (A$\cap$B)

    Anstatt nach und nach das gewünschte Objekt zu erstellen wird das Negativ konstruiert und von einer Grundform abgezogen. Zur Zeit übliche Vorgangsweise.

2.2 Parametric-associative design

16. Was ist parametrische Konstruktion
    P.2 F.31

    - parametrische Konstruktion verbindet geometrische Objekte mit geometrischen Beschränkungen und Maßangaben
    - Veränderungen der Geometrie durch Änderung der Eingabewerte der dazugehörigen Beschränkungen
    - Assoziative Konstruktionsoperationen schließen Beziehungen und Abhängigkeiten zwischen geometrischen Objekten direkt ein. Diese geometrischen Verbindungen sind als Teil des Kontruktionsprozesses definiert und erstellen Eltern-Kind Beziehungen oder Verhältnisse mehrerer Geometrien zum selben Parameter, zu den selben Parametern.
    - Moderne CAD Systeme bieten Mehr-Modell-Verknüpfungen, welche die Definition assoziativer Funktionalität zwischen (vormals unabhängigen) Teilen in Baugruppen anbietet.

17. Parametrische Beschreibung
    P.2 F.32

    - Virtuelle Produkt
      - Parameter
        - geometische Parameter (Koordinaten, Dimensionen, ...)
        - Material Parameter (Dichte, Festigkeit, Rauheit, ...)
        - Technologische Parameter (Vorschub, Schnittgeschwindigkeit, ...)
      - Beschränkungen
        - geometische Beschränkungen (absolute und relative Positionen, Orientierungen, ...)
        - Ingenieurliche(?) Beschränkungen (funktional / logisch)
      - strukturelle Parameter
        - Struktur der Montagegruppen

    - Struktur der Teile

18. Herausforderungen bei parametrischer Konstruktion
    P.2 F.43

    - Ingenieure tendieren zu unterschiedlicher Implementation von Methoden
    - parametrische Modelle werden komplexe, intrasparente Strukturen
    - Mehr-Modell-Verknüpfungen (Assoziationen zwischen Modellen) können zu Organisationsproblemen führen
    - erhöhtes Datenvolumen, vor allem bei unterschiedlichem Datenstand
    - komplexe Verknüpfungen führen zu umfangreicher Datenverarbeitung
    - komplexe Verknüpfungen führen zu „Referenzierung im Kreis“ → logische Probleme
    - parametrische Modelle enthalten Expertenwissen → Problem des Wissenstransfer

2.3 Knowledge based design

19. Was sind Wissensträger?
    P.2 F.46

    - Skizzen
    - Parameter, Beziehungen, Regeln, Schemen
    - Benutzer-features
    - Modell Verlauf
    - Vorlagen Modelle
    - Startup models
    - Produktstruktur
    - Attribute und Anmerkungen
    - DMU - Funktionalität (z. B. kinematische Simulation)
    - ...

20. Welche Arten von Wissensträger gibt es?
    P.2 F.47

    - Modelle mit fixer Geometrie
    - Modelle mit variabler Geometrie
    - Integration von mathematischen / logischen Beziehungen
    - automatisierte Abläufe für die geometrische Erzeugung oder berechnende Prozeduren
    - interaktive Programme

21. Was ist eine Template?

22. Levels of knowledge content in CAD models
    P.2 F.49
    Ansteigender Wissengehalt

Art des CAD Modells	Anwendungsbeispiel
1. neutrale Datenformate (IGES, STEP, etc.)	Virtualisierung
2. reduziertes Geometriemodelle (z. B. teilweise vereinfachte Geometrie)	DMU, Zulieferkette
3. detaillierte Geometriemodelle (z. B. isolierte Geometrie)	Zeichnungen, Simulationen
4. parametrische Modelle	Entwurf, Entwicklung
5. Nutzer features	Design-Prozess
6. Vorlagenmodelle	Fortgeschrittenes design
7. Skelett- und adaptive Modelle	Design in context

2.4 Assembling and product structures

23. Welche Elemente sind in Baumstruktur (Übersicht)
    P.2 F.51

    - Teil-Nummer / Name der Komponente
      - Axen Systeme
      - Parameter / Beziehungen
      - Eingabedaten / externe Geometrien
        * Styling Surface
        * Adapter Geometry
        * externe Begrenzungen, etc.
      - Referenzelemente
        * Referenzpunkte, -linien, -flächen
        * Referenzskizzen, -gitterelemente
      - Standards und Informationen
        * Demold
        * Informationen bzgl. Produktion
        * Anmerkungen, etc.
      - Definition der Geometrie
        * Part Geometrie
          · grundlegende Geometrie
          · Wellen/Furchen
          · Flanschen
          · Taschen
          · etc.
        * Schnittflächen
        * geschnittene Teilgeometrie
      - Veröffentlichung der finalen Oberfläche
      - Berechnungen und Maße
        * Gewicht, Fläche, etc.

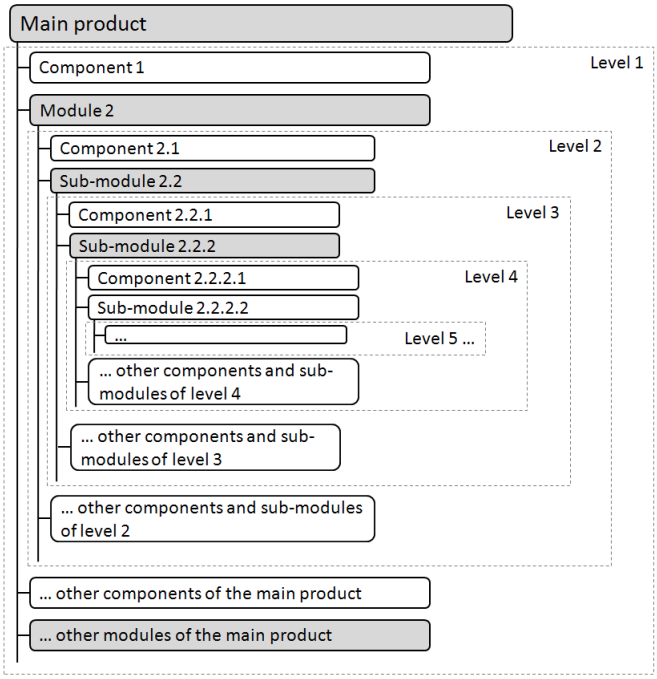
24. Wozu?

2.5 Aufbau und Produktstruktur

25. Was wird bei einer Baumgruppenkonstruktion gemacht?
    P.2 F.56
    Aufbaudesign beinhaltet die Organisation und Verknüpfung von zahlreichen Komponenten und Modulen von Produkten
    Aufbaudesign behandelt
    - Positionierung von Komponenten
    - Definition von Verhältnis und Verknüpfungen zwischen Komponenten
    - Organisation von Mehr-Modell-Verknüpfungen
    - Organisation von Parameterstrukturen
    - Bereitstellung von Geometrie und Struktur für nachfolgende DMU-Prozesse
    - Bereitstellung von Geometriedaten für Montagebasierte-Simulation (z. B. MBS)
    - Interaktion mit PDM-Systemen

26. Wie werden komplexe Baugruppen organisiert?
    P.2 F.59

Einteilung in Hierarchie-Ebenen



27. Methoden der Positionierung
P.2 F.61

    (a) Durch Beschränkungen

    (b) Durch ein oder mehrerer Skelettmodelle

    (c) Relativ zu einem Referenzkoordinatensystem

3 Virtuelle Entwicklung mechatronischer Produkte

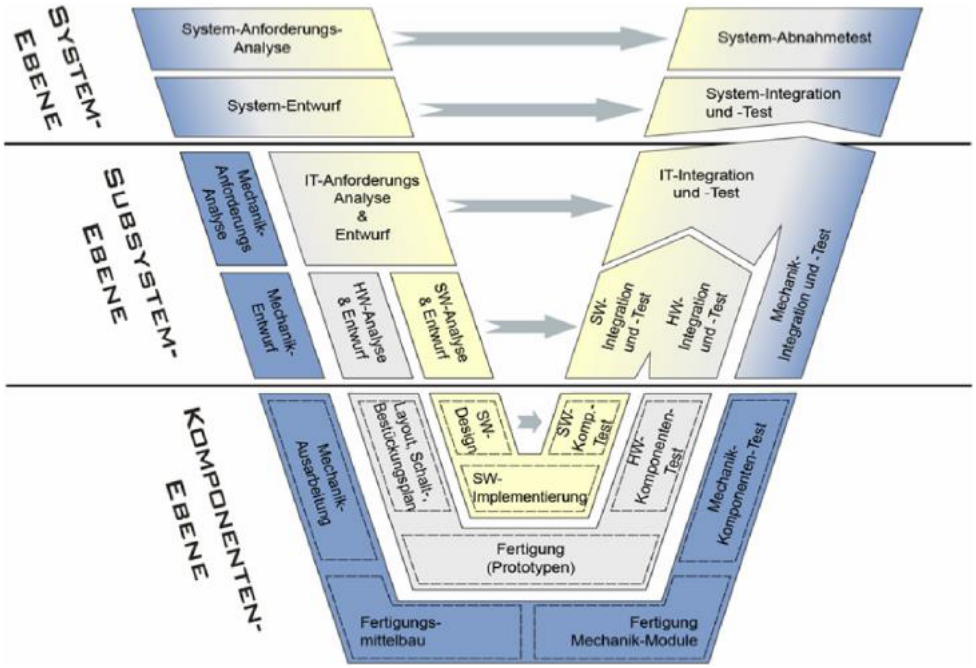
3.1 Einleitung - Mechatronik

28. Was ist Mechatronik?
P.3 F.3-4
Die Mechatronik beschreibt technische Systeme, welche sich aus mechanischen, elektrischenund elektronischen Komponenten zusammensetzen.

29. Randbedingung, was ist kritisch
P.3 F.8

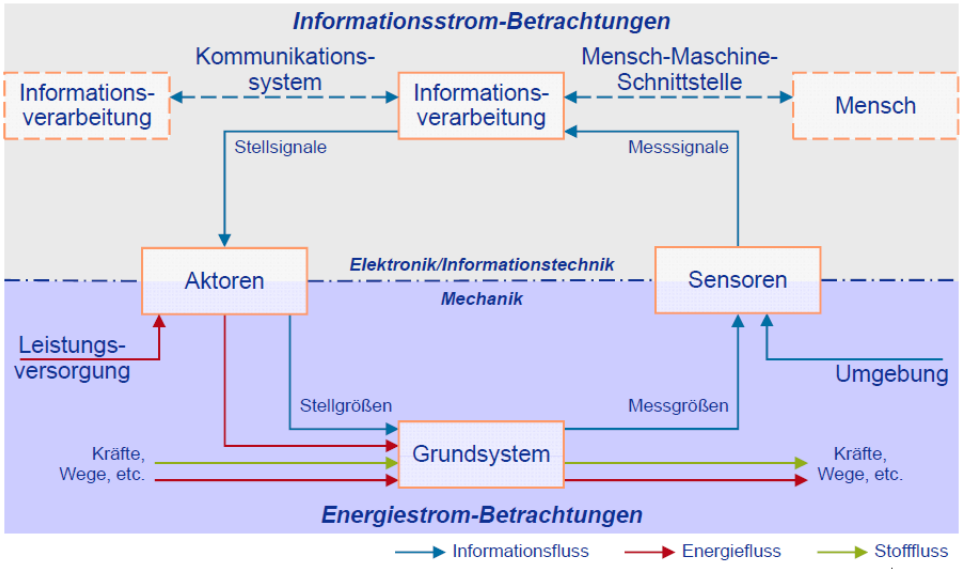
Äußere Randbedingungen	Innere Randbedingungen
kurze Lebensdauer mechatronischer Produkte/ schneller Modellwechsel	fehlende domäneübergreifende Spezifikation der Produkte (Beschreibungssprache)
hohe Zahl von Varianten (Kundenwünsche)	unterschiedliche Entwicklungsmethoden und IT Werkzeuge in den Domänen
Produkte müssen zuverlässiger werden (Wettbewerb)	Fehlende gemeinsame Simulatino der zu entwickelnden Produkte
Entwicklungskosten müssen gesenkt werden (Wettbewerb)	Hohe Produktkomplexität

30. V-Modell!
P.3 F.15



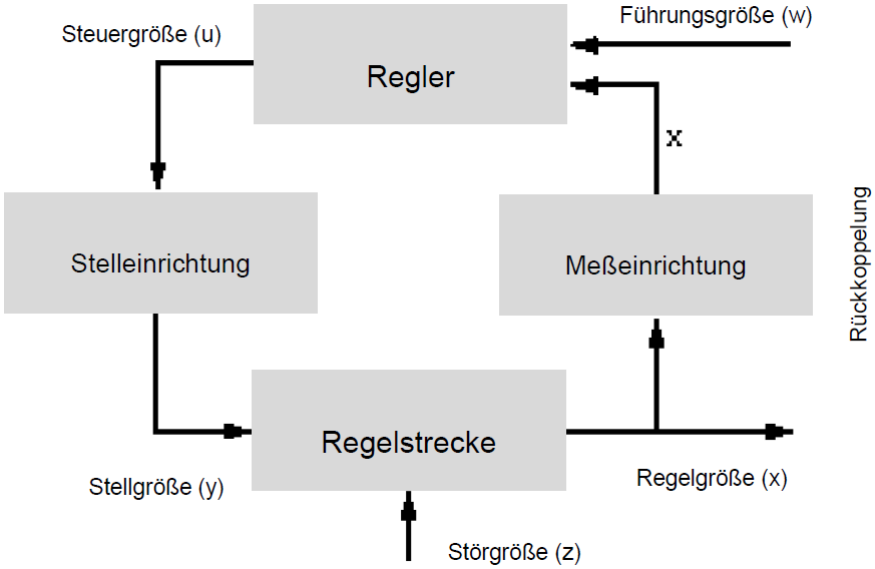
3.2 Komponenten mechatronischer Systeme

31. Übersicht Komponenten



P.3 F.17

32. Aufbau Regelkreis
P.3 F.18



33. Definition Aktor, Sensor, Prozessdatenverarbeitung

- Aktoren: P.3 F.19-23
  Technisch gesehen versteht man unter einem Aktor die Zusammenschaltung eines Energiewandlers mit einem Leistungsstellglied.
  Das Leistungsstellglied verbindet die eingehende Energie (in der Regel elektrische Energie) mit dem Stellsignal. Es entsteht eine modulierte Energie, die vom Wandler in die Energieart der Stellgröße (meist mechanische Energie) transformiert wird.
  In mechatronischen Systemen stellen Aktoren das Bindeglied zwischen der Informationsverarbeitung und dem mechanischen Grundsystem dar.
- Sensoren: P.3 F.24-28
  Sensoren wandeln die zu messenden physikalischen oder chemischen Größen in elektrische Signale um.
  Ein Signal stellt in diesem Sinne eine zeitvariable, physikalische oder chemische Zustandsgröße (z.B. Druck, Temperatur, Kraft, usw.) als zeitliche Abfolge von Messwerten dar.
  Funktional besteht ein Sensor aus
  - einem Sensorelement, das die Messgröße in ein elektrisches Signal umwandelt, und
  - einer Signalverarbeitung, die ein genormtes elektrisches Ausgangssignal liefert

  Somit sind Sensoren Komponenten, welche ein im allgemeinen nicht elektrisches Signal (Messignal) in ein elektrisches Signal umwandeln.
- Prozessdatenverarbeitung: P.3 F.30
  Die Verarbeitung von Prozessdaten erfolgt in den meisten Fällen in Mikroprozessoren. Hier werden in Abhängigkeit der Komplexität der mechatronischenSysteme verschiedene Anforderungen gestellt.
  Echtzeitdatenverarbeitung: Systeme sind echtzeitfähig wenn sie in der Lage sind, unabhängig von Art und Umfang des gerade bearbeiteten Problems auf ein zu beliebiger Zeit auftretendes Ereignis höherer Dringlichkeit spätestens nach Ablauf einer definierbaren maximalen Reaktionszeit t_{Rmaxin} programmierbarer Weise zu reagieren. Die Reaktionszeit bezeichnet dabei die Zeit, die der Rechner benötigt bis der Task gerechnet werden kann.
  Multitasking: Dieses Verfahren ermöglichen die Berücksichtigung mehrerer Prozesse/ Threads durch den Prozessor. Die Verwaltung einer hohen Zahl von Tasks muss für die höchstpriorisierten Tasks ohne Auswirkungen bleiben. Synchronisationsmechanismen, die den Tasks erlauben, ihre Aktivitäten gegenseitig abzustimmen, müssen optimiert werden.
  Multiprocessing: Es gibt 3 Kategorien von Multiprocessing
  - Heterogene Architekturen: Die Prozessoren übernehmen verschiedenartige Aufgaben und sind oft auch nicht vom gleichen Typ.
  - Lose gekoppelte Architekturen: Die Prozessoren besitzen eigene Speicher, sind jedoch über schnelle Datenverbindungen gekoppelt.
  - Eng gekoppelte Architekturen: Die Prozessoren sind gleichartig und teilen sich auch den Speicherbereich.

3.3 Hardware in the Loop (HiL) / Software in the Loop (SiL)

34. Was ist SiL/HiL, wie wird es angewandt.
P.3 F.32-37

- **Hardware in the Loop (HiL)** Hardware in the Loop (HIL) ist eine Methode zum Testen und Absichern von eingebetteten Systemen, zur Unterstützung während der Entwicklung sowie zur vorzeitigen Inbetriebnahme von Maschinen und Anlagen.

  Bei Hardware in the Loop (HIL) wird ein reales System (z.B. reales elektronisches Steuergerät oder reale mechatronische Komponente) über seine Ein- und Ausgänge an ein angepasstes Gegenstück (HIL-Simulator) angeschlossen. Dabei übernimmt der HIL-Simulator die Nachbildung der realen Umgebung des Systems durch virtuelle Methoden.

  Bei HIL für eingebettete Systeme wird das zu steuernde System (z.B. Auto) über Modelle simuliert, um die korrekte Funktion des zu entwickelnden Steuergerätes (z.B. Motorsteuergerät) zu testen. Die HIL-Simulation muss meist in Echtzeit ablaufen und wird in der Entwicklung benutzt, um Entwicklungszeiten zu verkürzen und Kosten zu sparen.

  Insbesondere lassen sich wiederkehrende Abläufe simulieren. Dies hat den Vorteil, dass eine neue Entwicklungsversion unter den gleichen Kriterien getestet werden kann wie die Vorgängerversion. Somit kann detailliert nachgewiesen werden, ob ein Fehler beseitigt wurde oder nicht.

  Der HIL-Simulator besteht also aus einem Rechner, der die Echtzeitbedingungen der jeweiligen Anwendung erfüllen kann (zunehmend auch PC-basiert), digitale und analoge Ein-und Ausgabe-Schnittstellen zum Steuergerät und Ersatzlasten, die der steuergeräteinternen Endstufendiagnose simulieren, dass alle Aktuatoren korrekt angeschlossen seien.

  Die Tests an realen Systemen lassen sich dadurch stark verringern und zusätzlich lassen sich Systemgrenzen ermitteln, ohne das Zielsystem (z.B. Auto und Fahrer) zu gefährden.

  Die HIL-Simulation ist immer nur eine Vereinfachung der Realität und kann den Test am realen System deshalb nicht ersetzen. Falls zu große Diskrepanzen zwischen der HIL- Simulation und der Realität auftreten, sind die zugrundeliegenden Modelle in der Simulation zu stark vereinfacht. Dann müssen die Simulations-Modelle weiterentwickelt werden.

  Die Entwicklung mittels HIL ist mittlerweile in verschiedenen Branchen der mechatronischen Produktentwicklung vertreten, z.B. in der Automobilindustrie, der Luft- und Raumfahrt, sowie im Maschinen- und Anlagenbau.

  Beispiel für eine angewandte HIL-Entwicklungsmethodik in der Automobilindustrie. Das HIL-System simuliert dabei die Umgebung (Fahrzeug, Reifen, Straße) des realen Steuergerätes.

- **Software in the Loop (SiL)** Bei der Methode Software in the Loop (SIL) wird im Gegensatz zum HIL keine besondere Hardware eingesetzt. Das erstellte Modell der Software wird lediglich in den für die Zielhardware verständlichen Code umgewandelt (beispielsweise von einem MATLAB/ Simulink-Modell). Dieser Code wird auf dem Entwicklungsrechner zusammen mit dem simulierten Modell ausgeführt, anstatt wie bei Hardware in the Loop auf der Zielhardware zu laufen. Es handelt sich dabei also um eine Methode, die vor dem HIL anzuwenden ist.

  Vorteile von SIL sind unter anderem, dass die Zielhardware noch nicht feststehen muss, und dass die Kosten aufgrund der fehlenden Simulationsumgebung weitaus geringer ausfallen. Das hier benutzte Modell der Strecke kann auch beim HIL weiter verwendet werden, und somit die einzelnen Testläufe miteinander verglichen werden.

  Nachdem sowohl der Prozess, als auch die Regelung / Steuerung in der Simulation vorliegen, sind verschiedene Testszenarien denkbar. Wird der Regelungsalgorithmus unter Verwendung einer geeigneten Codegenerierung auf eine leistungsfähige Simulations-Hardware übertragen, kann der reale Prozess mit dem Regelungsalgorithmus betrieben werden, ohne, dass bereits an dieser Stelle ein Mikrokontroller oder Ähnliches mit den einhergehenden Nachteilen (Speicherplatzbeschränkung, begrenzte Prozessorleistung) eingesetzt werden müsste → „Software in the Loop“.

3.4 Computer aided software engineering (CASE)

35. Upper-/Lower CASE
P.3 F.39-41 Unter Computer-aided software engineering (CASE) versteht man den intensiven Einsatz IT-gestützter Werkzeuge für die Umsetzung einer Software-Konzeption. Ziel ist es, Software möglichst vollständig automatisiert aus fachlichen Beschreibungen (Vorgeben) zu erstellen.

CASE-Tools sind Programme, die den Software-Ingenieur bei der Planung, dem Entwurf und der Dokumentation seiner Arbeitsergebnisse (Software) unterstützen. Ein wichtiger Bestandteil von CASE-Tools ist eine grafische Notationsweise, die der Visualisierung der Architektur des Software-Systems dient.

CASE-Tools sind oft in Entwicklungsumgebungen (IDEs) integriert; manchmal sind es auch eigenständige Applikationen, deren Fokus vollständig auf CASE liegt (ohne dabei die anderen typischen Elemente einer Entwicklungsumgebung anzubieten).

Upper-CASE-Tools unterstützen die frühen (oberen) Phasen der IS-Entwicklung. Sie erlauben die rechnergestützte konzeptionelle Modellierung, beispielsweise mit Hilfe von Entity-Relationship-, Funktionshierarchie-, Datenflußdiagrammen usw.

Lower-CASE-Tools bestehen aus einem Satz von Programmgeneratoren, die aufgrund der erfassten Modelldaten selbständig den benötigten Programmcode für die gewünschten Anwendungen erzeugen.