Panel:

Challenges in Complex Software Systems

(requirements, adaptation, updates, real-time, user behavior, deployment, etc.)

SoftNet 2020

Chair

Herwig Mannaert, University of Antwerp, Belgium

Panelists

Dieter Landes, University of Applied Sciences Coburg, Germany

Frank Herrmann, OTH Regensburg, Germany

Radek Koci, Brno University of Technology, Czech Republic

Kevin Lano, King's College London, United Kingdom

Panel:
Challenges in Complex Software Systems
(requirements, adaptation, updates, real-time, user behavior, deployment, etc.)

SoftNet 2020

Panellist Positions

Sustainable and evolvable traceability requires advanced manufacturing

Herwig Mannaert, University of Antwerp, Belgium, herwig.mannaert@uantwerp.be

Software Engineering Education Matters!

Dieter Landes, University of Applied Sciences Coburg, Germany dieter.lands@hs-coburg.de

Planning for Industrial Practice needs to be Generic!

Frank Herrmann, OTH Regensburg, Frank.Herrmann@OTH-Regensburg.de.

From Requirements to Implementation and Back

Radek Koci, Brno University of Technology, Czech Republic, koci@fit.vut.cz

Generating complex mobile applications

Kevin Lano, King's College London, United Kingdom, kevin.lano@kcl.ac.uk

Panel:
Challenges in Complex Software Systems
(requirements, adaptation, updates, real-time, user behavior, deployment, etc.)

SoftNet 2020

Panelists

Herwig Mannaert, University of Antwerp, Belgium *(Chairman)*

Dieter Landes, University of Applied Sciences Coburg, Germany

Frank Herrmann, OTH Regensburg, Germany

Radek Koci, Brno University of Technology, Czech Republic

Kevin Lano, King's College London, United Kingdom

Panel:
Challenges in Complex Software Systems
(requirements, adaptation, updates, real-time, user behavior, deployment, etc.)

SoftNet 2020

Panellist Position

Software Engineering Education Matters!

Dieter Landes, University of Applied Sciences Coburg, Germany dieter.lands@hs-coburg.de

- Complex (software) systems require skilled developers and engineers
- Complexity of systems makes SE education more challenging
    - New stakeholders
    - Increasing transdisciplinarity
    - New technologies

→ Competences, not just knowledge

→ Working in transdisciplinary teams with stakeholders with diverse backgrounds

→ Sound didactical basis, not just fancy educational technology

→ Transdisciplinary approach for developing didactical concepts in SE education

IARIA

Panel:
Challenges in Complex Software Systems
(requirements, adaptation, updates, real-time, user behavior, deployment, etc.)

SoftNet 2020

Panellist Position

Planning for Industrial Practice needs to be Generic!

Frank Herrmann, OTH Regensburg, Frank.Herrmann@OTH-Regensburg.de.
Innovation and Competence Centre for Production Logistics and Factory Planning.

Requirements in industrial practice:

- Specially-designed production systems.
- Additional consideration of workers, maintenance, stochastics and more .
- Change of production systems from time to time.

⇒ High demands on high-performance planning algorithms.

Proposal: Solving Models.

Panel:
Challenges in Complex Software Systems
(requirements, adaptation, updates, real-time, user behavior, deployment, etc.)

SoftNet 2020

Panellist Position

From Requirements to Implementation and Back

Radek Koci, Brno University of Technology, Czech Republic, koci@fit.vut.cz

- Complex set of changing requirements
- Very difficult to keep track of requirements and their implementation
- Changes in requirements or implementation have to be traceable and interconnected
- Formal models for requirement specification and analysis

✓ Models and implementations are inextricably linked

✓

Panel:

Challenges in Complex Software Systems

(requirements, adaptation, updates, real-time, user behavior, deployment, etc.)

SoftNet 2020

Panellist Position

Generating complex mobile applications

Kevin Lano, King's College London, UK kevin.lano@kcl.ac.uk

- Diversity of mobile platforms and versions (eg., Android, iOS/UIKit, iOS/SwiftUI)
- Rapid evolution of mobile platforms, languages and ecosystems
- Production of efficient and usable apps
- Interaction with cloud and remote services
- Sophisticated use of machine learning for activity detection/personalisation

→ Manual coding and maintenance of apps is too expensive & slow

→ Automated synthesis of apps from platform-independent specifications

→ Lightweight code-generation technology to support rapid evolution of mobile platforms

→ Future work will incorporate strategies for using machine learning techniques

IARIA

Panel:
Challenges in Complex Software Systems
(requirements, adaptation, updates, real-time, user behavior, deployment, etc.)

SoftNet 2020

Panellist Position

Sustainable and evolvable traceability requires advanced manufacturing

Herwig Mannaert, University of Antwerp, Belgium, herwig.mannaert@uantwerp.be

- Requirements change and will continue to evolve
- We need to achieve traceability between requirements and their implementation
- Traceability needs to be sustained while requirements continue to arrive and evolve
- Traceability needs to be sustained while migrating to new underlying technologies
- Contemporary modeling and implementation techniques do not provide this

➔ ***We require advanced implementation techniques similar to industrial manufacturing***

PANEL:
Challenges in Complex Software Systems

SoftNet 2020

Prof. dr. Herwig Mannaert

Universiteit Antwerpen

Sustainable and evolvable traceability requires prescriptive design rules and advanced manufacturing techniques

On “Design Science”

- "The function of what I call design science is to solve problems by introducing into the environment new artifacts, the availability of which will induce their spontaneous employment by humans and thus, coincidentally, cause humans to abandon their previous problem-producing behaviors and devices. For example, when humans have a vital need to cross the roaring rapids of a river, as a design scientist I would design them a bridge, causing them, I am sure, to abandon spontaneously and forever the risking of their lives by trying to swim to the other shore."

- R. Buckminster Fuller, from *Cosmography*

On Architecture

- "**Architecture** (Latin *architectura*, from the Greek ἀρχιτέκτων *arkhitekton* "architect", from ἀρχι- "chief" and τέκτων "creator") is both the process and the product of planning, designing, and constructing buildings or any other structures.[3] Architectural works, in the material form of buildings, are often perceived as cultural symbols and as works of art. Historical civilizations are often identified with their surviving architectural achievements."

— Wikipedia

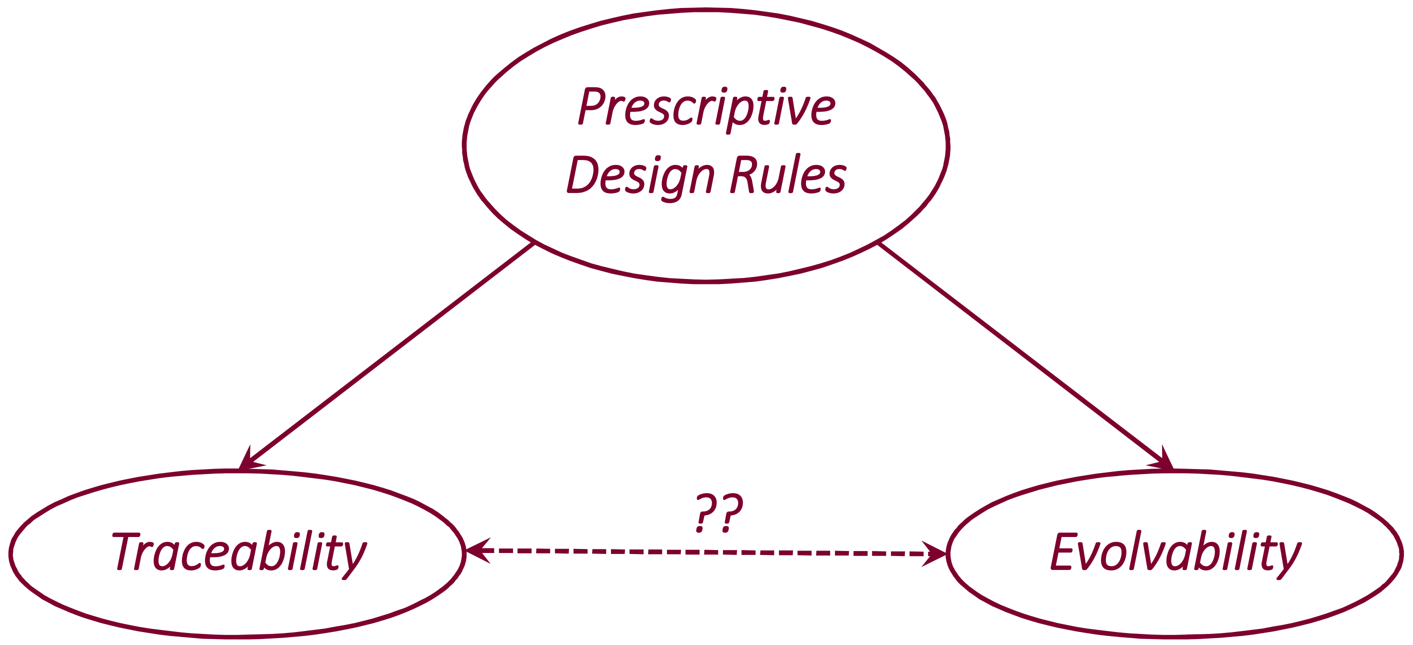
On Software Architecture

- **Software architecture** refers to the fundamental structures of a software system and the discipline of creating such structures and systems. Each structure comprises software elements, relations among them, and properties of both elements and relations.[1] The *architecture* of a software system is a metaphor, analogous to the architecture of a building.[2] It functions as a blueprint for the system and the developing project, laying out the tasks not necessary to be executed by the design teams.[3]
- Software architecture is about making fundamental structural choices that are costly to change once implemented. Software architecture choices include specific structural options from possibilities in the design of software. For example, the systems that controlled the space shuttle launch vehicle had the requirement of being very fast and very reliable. Therefore, an appropriate real-time computing language would need to be chosen. Additionally, to satisfy the need for reliability the choice could be made to have multiple redundant and independently produced copies of the program, and to run these copies on independent hardware while cross-checking results.
- Documenting software architecture facilitates communication between stakeholders, captures early decisions about the high-level design, and allows reuse of design components between projects

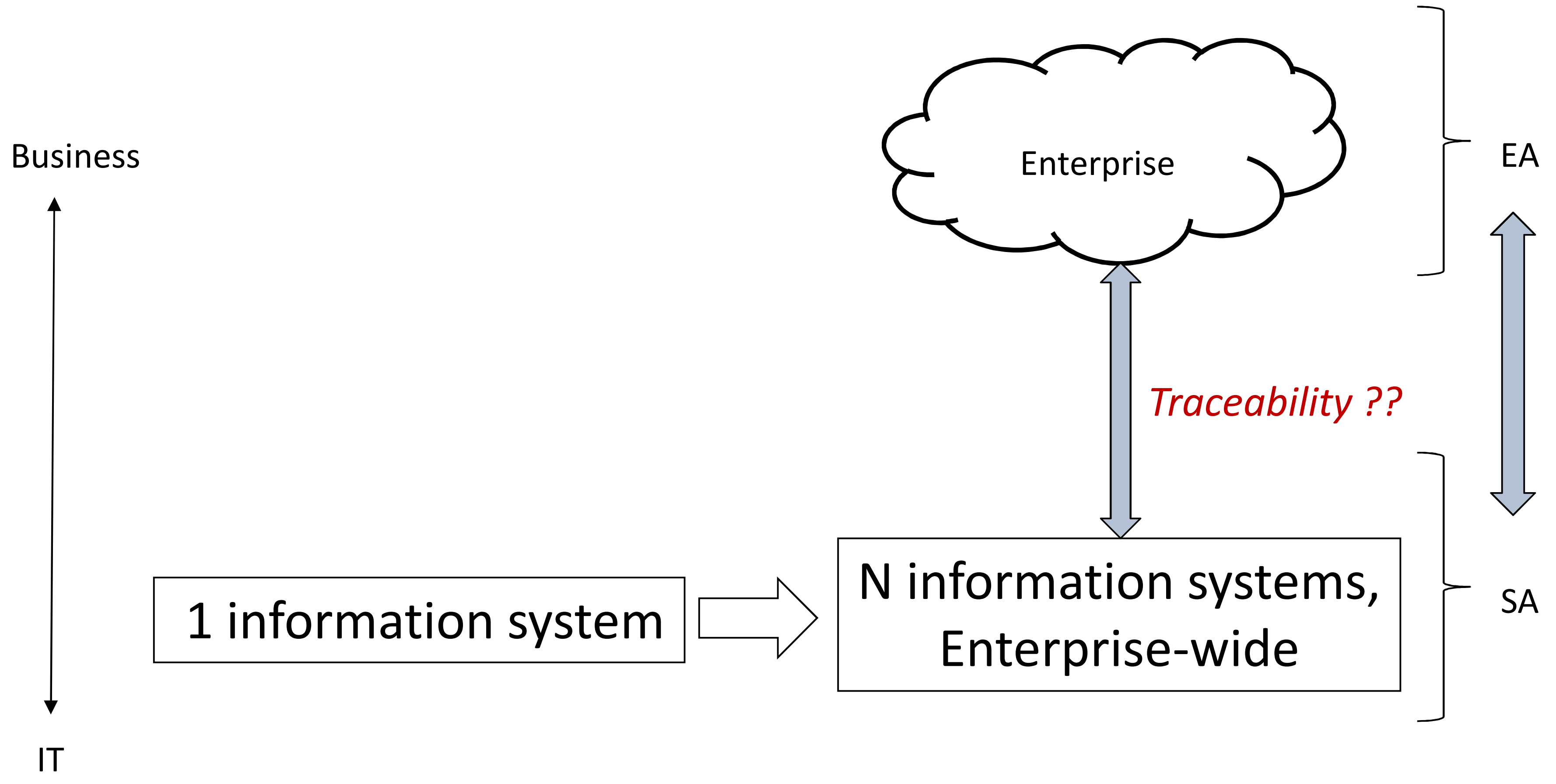
Scope of Software Architecture

- *Overall, macroscopic system structure*;[6] this refers to architecture as a higher level abstraction of a software system that consists of a collection of computational *components* together with *connectors* that describe the interaction between these components.
- *The important stuff—whatever that is*;[7] this refers to the fact that software architects should concern themselves with those decisions that have high impact on the system and its stakeholders.
- *That which is fundamental to understanding a system in its environment"*[8]
- *Things that people perceive as hard to change*;[7] since designing the architecture takes place at the beginning of a software system's lifecycle, the architect should focus on decisions that "have to" be right the first time. Following this line of thought, architectural design issues may become non-architectural once their irreversibility can be overcome.
- *A set of architectural design decisions*;[9] software architecture should not be considered merely a set of models or structures, but should include the decisions that lead to these particular structures, and the rationale behind them. This insight has led to substantial research into software architecture knowledge management.[10]

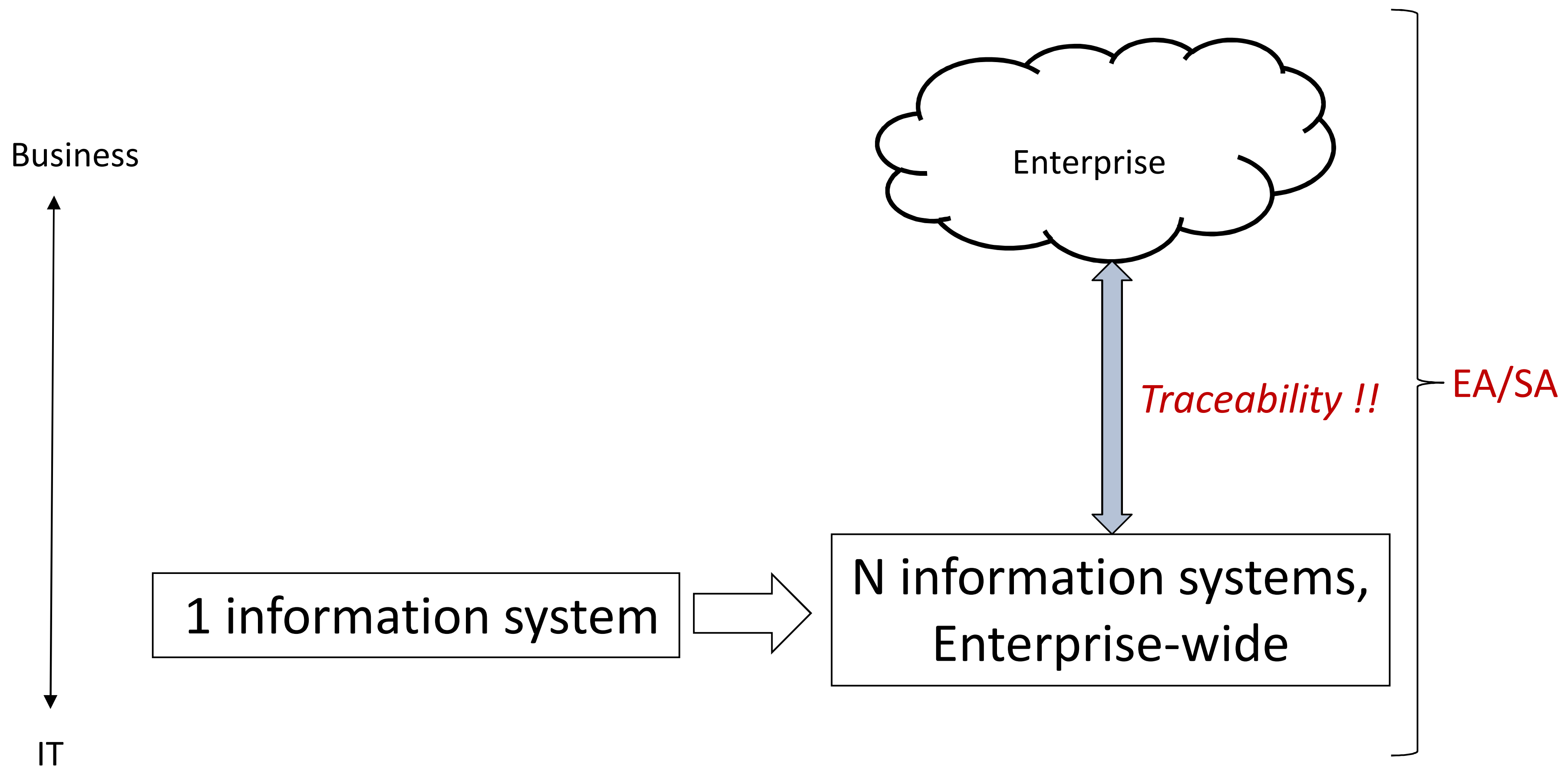
Crucial Issues for Software Architecture



Software Architecture and Need for Traceability



Software Architecture and Need for Traceability



Generating complex mobile applications

Kevin Lano, King's College London, UK kevin.lano@kcl.ac.uk

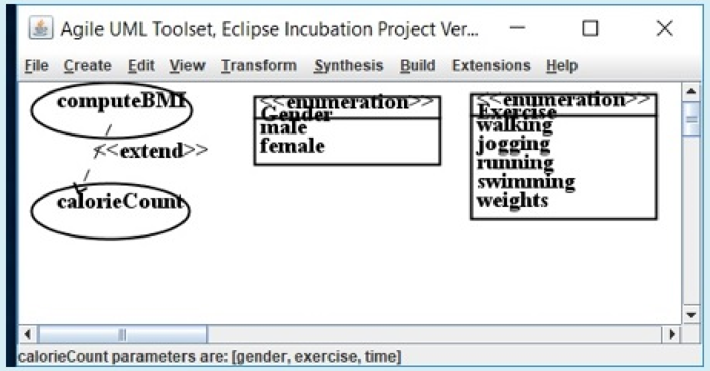
KING'S College LONDON

University of London

- Both Android and iOS are complex platforms, supporting UI elements, UI interaction, resource management, asynchronous execution, service activation, etc
- Manual development in either or both platforms involves substantial effort & high expertise to produce usable and commercially successful apps
- Work in Model-driven-engineering (MDE) aims to automate app production, using graphical/text models & automated code generation from these.

iOS and Android

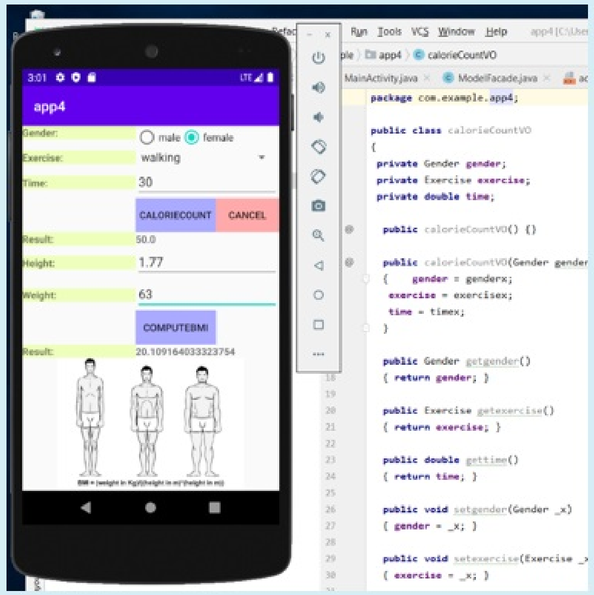
Many common aspects – but different code details. It makes sense to write a single app specification & auto-generate code for the platforms



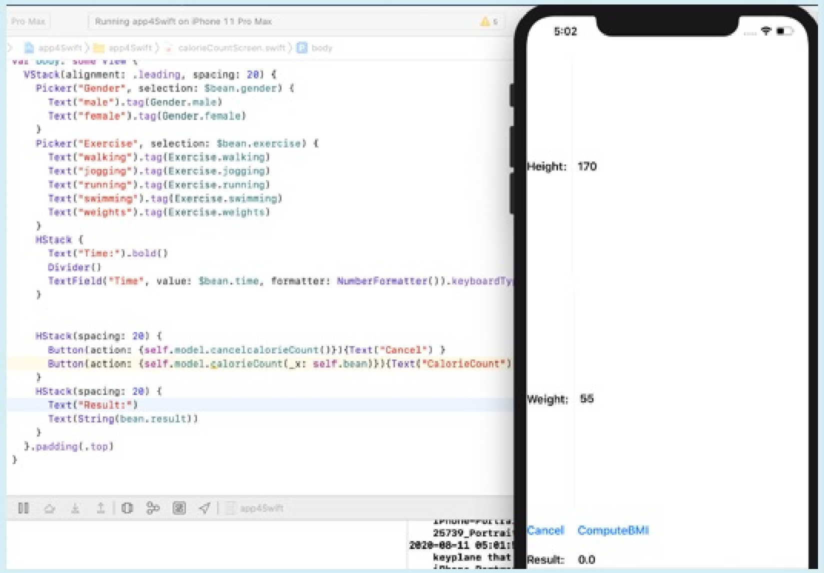
calorieCount parameters are: [gender, exercise, time]

Android Java

iOS SwiftUI



```
package com.example.app4;

public class calorieCountVO
{
 private Gender gender;
 private Exercise exercise;
 private double time;

  public calorieCountVO() {}

  public calorieCountVO(Gender gender
  {    gender = genderx;
   exercise = exercisex;
   time = timex;
  }

  public Gender getgender()
  { return gender; }

  public Exercise getexercise()
  { return exercise; }

  public double gettime()
  { return time; }

  public void setgender(Gender _x)
  { gender = _x; }

  public void setexercise(Exercise _x
  { exercise = _x; }
```

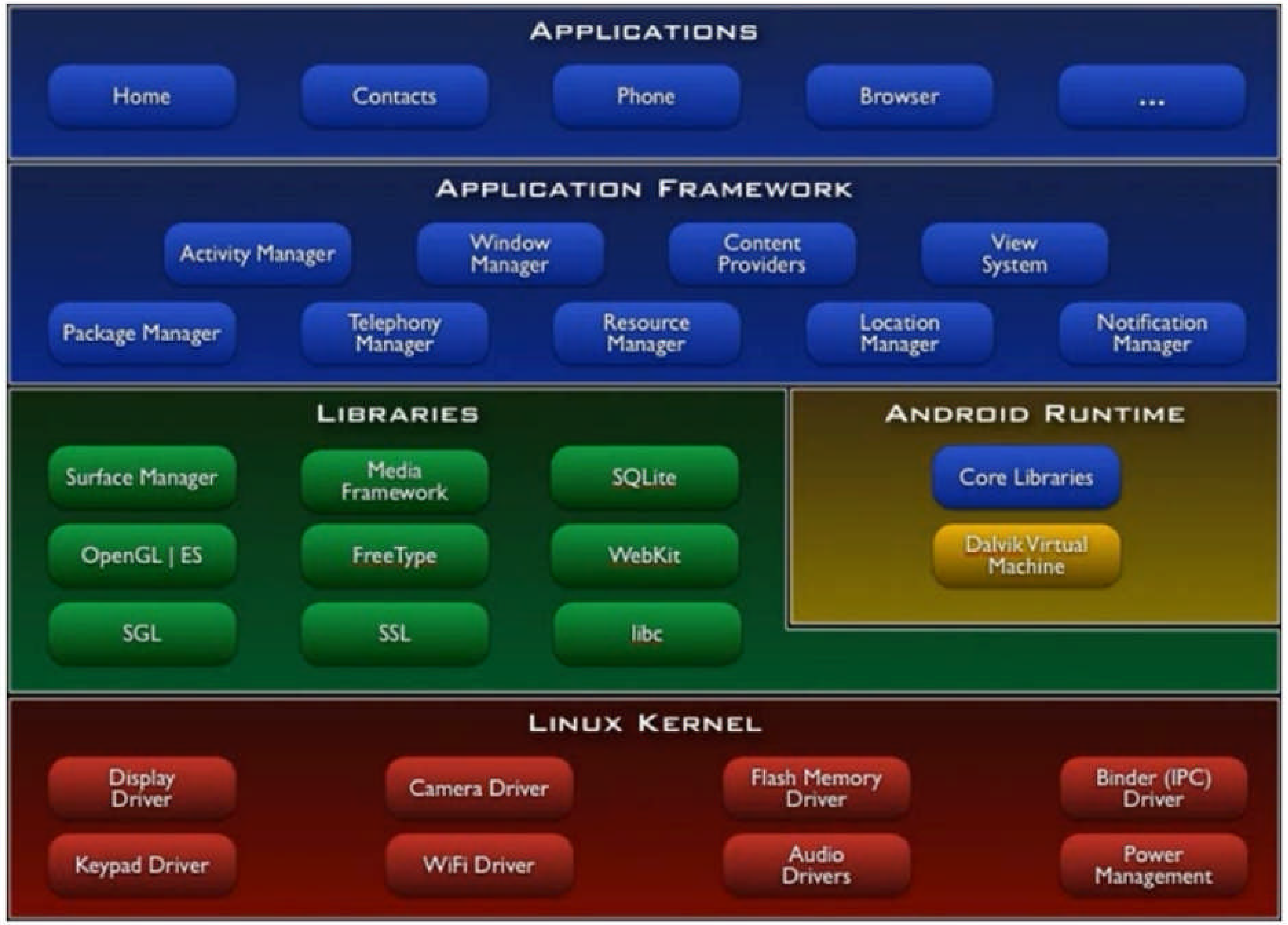


```
  VStack(alignment: .leading, spacing: 20) {
    Picker("Gender", selection: $bean.gender) {
      Text("male").tag(Gender.male)
      Text("female").tag(Gender.female)
    }
    Picker("Exercise", selection: $bean.exercise) {
      Text("walking").tag(Exercise.walking)
      Text("jogging").tag(Exercise.jogging)
      Text("running").tag(Exercise.running)
      Text("swimming").tag(Exercise.swimming)
      Text("weights").tag(Exercise.weights)
    }
    HStack {
      Text("Time:").bold()
      Divider()
      TextField("Time", value: $bean.time, formatter: NumberFormatter()).keyboardTy
    }


    HStack(spacing: 20) {
      Button(action: {self.model.cancelcalorieCount()}){Text("Cancel") }
      Button(action: {self.model.calorieCount(_x: self.bean)}){Text("CalorieCount")
    }
    HStack(spacing: 20) {
      Text("Result:")
      Text(String(bean.result))
    }
  }.padding(.top)
}
```

Mobile Applications

- **Diverse range of systems: business, entertainment, social, health**
  - Restricted interfaces (size), processing power & connectivity
  - Wide range of data sources
  - Usability, security, responsiveness are important QoS properties
  - Managing offline/online status
- **Multiple mobile device platforms exist**
  - Android and iOS are principal platforms
  - But many variants, and rapid evolution of languages (Java, Kotlin, Swift, SwiftUI, etc)

Android Architecture



Mobile application synthesis

- **Specification**
  - Text/graphical specification of data (classes), functionality (use cases, operations)
- **Design**
  - Utilise standard mobile patterns: model-view-controller, VIPER, MVVM, Service Activator, etc
- **Implementation**
  - Either Java for Android, or Swift or SwiftUI for iOS
  - UI elements, UI control code, business tier code and resource management code generated automatically from specification & design
  - Typically code is 10 times larger than specification

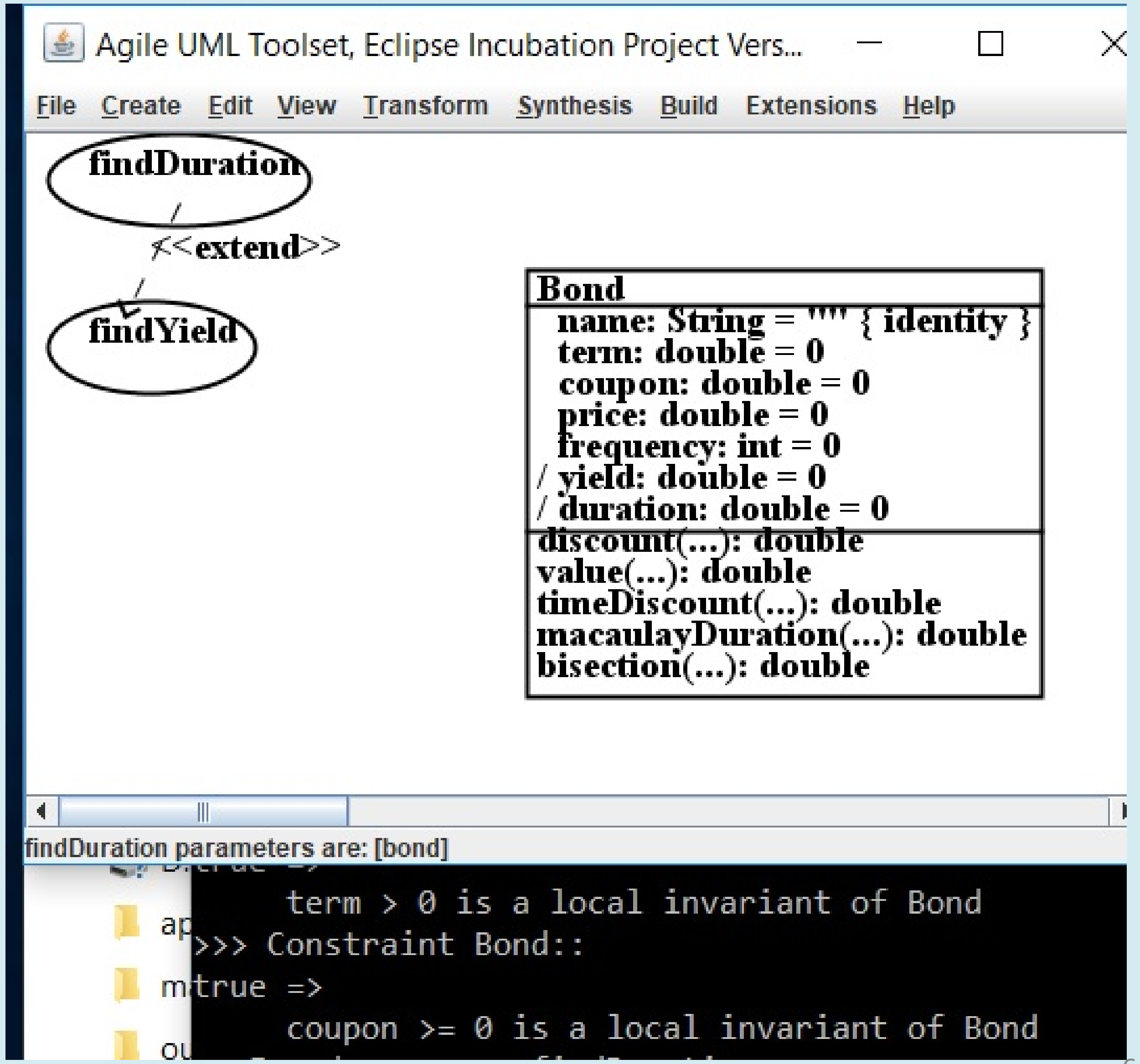
Example: bond analysis app

Specification in UML notation

Constraints on data will produce validation checks & test cases

Use cases become UI screens

Persistent classes become local or remote persistent data

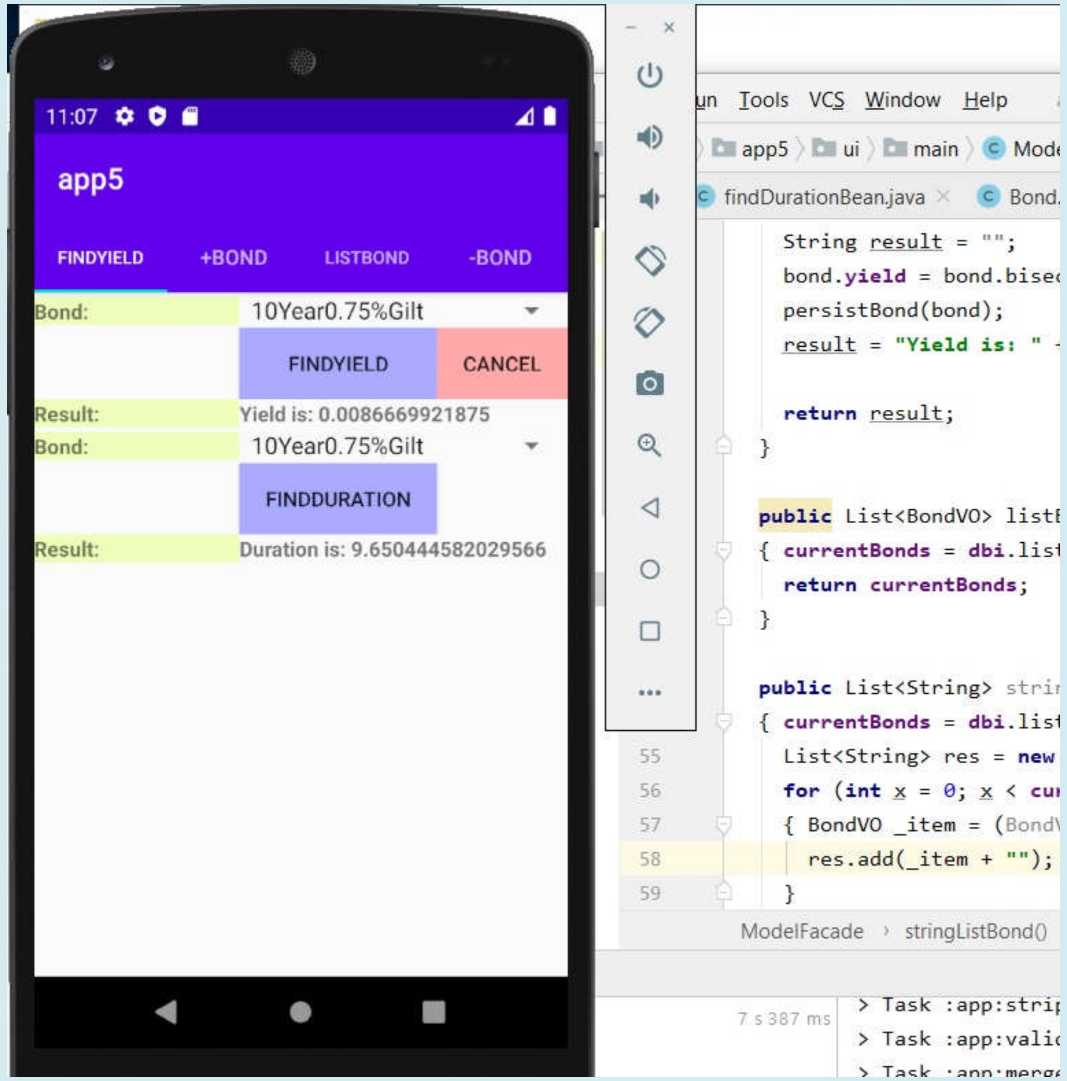

Agile UML Toolset, Eclipse Incubation Project Vers...
File Create Edit View Transform Synthesis Build Extensions Help
findDuration
<<extend>>
findYield

```
Bond
  name: String = "" { identity }
  term: double = 0
  coupon: double = 0
  price: double = 0
  frequency: int = 0
/ yield: double = 0
/ duration: double = 0
discount(...): double
value(...): double
timeDiscount(...): double
macaulayDuration(...): double
bisection(...): double
```

findDuration parameters are: [bond]

```
    term > 0 is a local invariant of Bond
>>> Constraint Bond::
true =>
    coupon >= 0 is a local invariant of Bond
```

Example: bond analysis app



```
11:07
app5
FINDYIELD   +BOND   LISTBOND   -BOND
Bond:    10Year0.75%Gilt
         FINDYIELD   CANCEL
Result:  Yield is: 0.0086669921875
Bond:    10Year0.75%Gilt
         FINDDURATION
Result:  Duration is: 9.650444582029566
```

```
    String result = "";
    bond.yield = bond.bisec
    persistBond(bond);
    result = "Yield is: " +

    return result;
  }

  public List<BondVO> listB
  { currentBonds = dbi.list
    return currentBonds;
  }

  public List<String> strin
  { currentBonds = dbi.list
    List<String> res = new
    for (int x = 0; x < cur
    { BondVO _item = (BondV
      res.add(_item + "");
    }
```

```
7 s 387 ms   > Task :app:strip
             > Task :app:valic
             > Task :app:merge
```

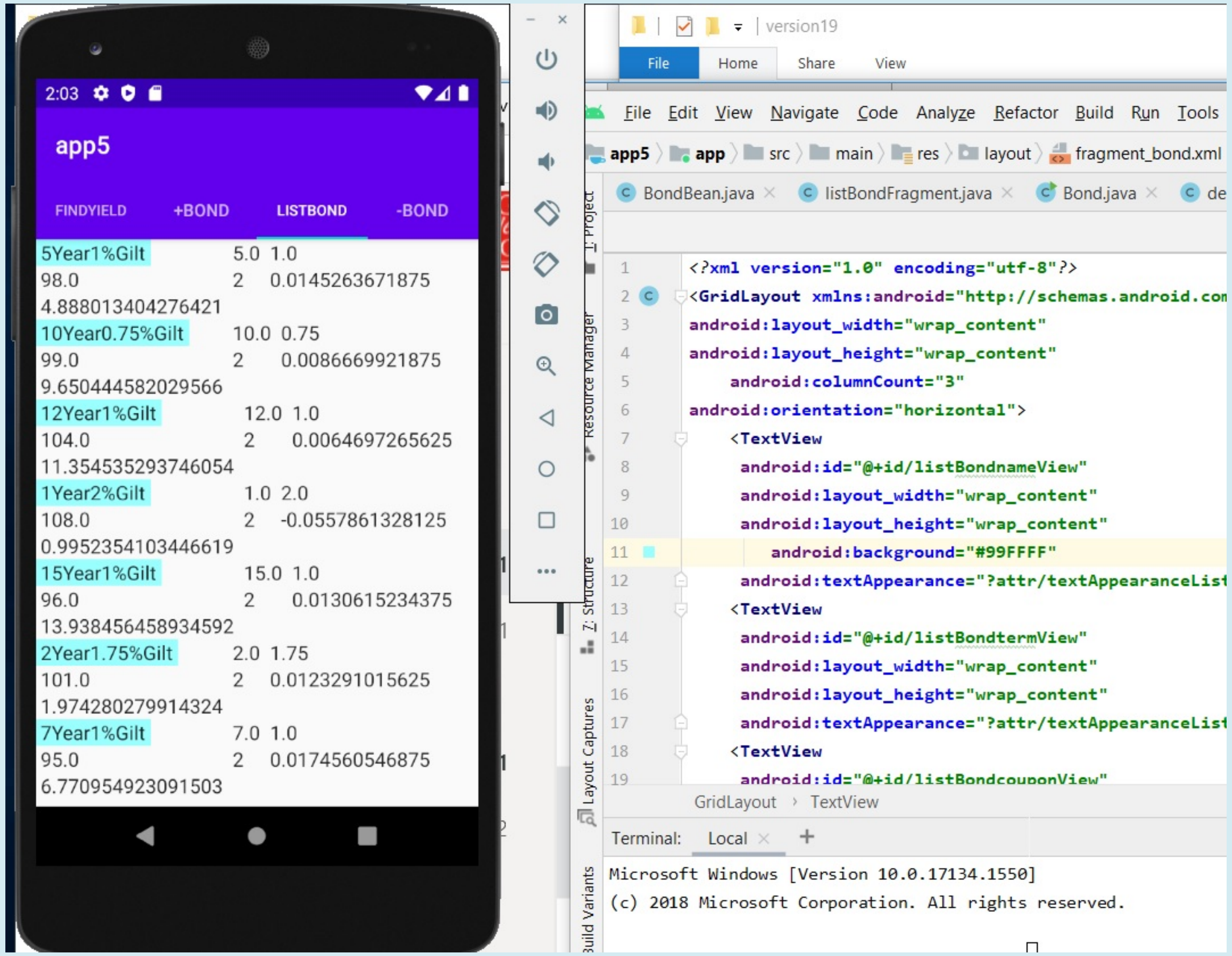
User can enter data about bonds (their term, coupon, etc).

App computes yield and duration.

Bonds can be listed and deleted.

Persistent data stored using SQLlite.

Bond analysis app



```
5Year1%Gilt          5.0  1.0
98.0                 2    0.0145263671875
4.888013404276421
10Year0.75%Gilt      10.0  0.75
99.0                 2     0.0086669921875
9.650444582029566
12Year1%Gilt         12.0  1.0
104.0                2     0.0064697265625
11.354535293746054
1Year2%Gilt          1.0  2.0
108.0                2    -0.0557861328125
0.9952354103446619
15Year1%Gilt         15.0  1.0
96.0                 2     0.0130615234375
13.938456458934592
2Year1.75%Gilt       2.0  1.75
101.0                2    0.0123291015625
1.974280279914324
7Year1%Gilt          7.0  1.0
95.0                 2    0.0174560546875
6.770954923091503
```

```xml
<?xml version="1.0" encoding="utf-8"?>
<GridLayout xmlns:android="http://schemas.android.com
android:layout_width="wrap_content"
android:layout_height="wrap_content"
    android:columnCount="3"
android:orientation="horizontal">
    <TextView
     android:id="@+id/listBondnameView"
     android:layout_width="wrap_content"
     android:layout_height="wrap_content"
        android:background="#99FFFF"
     android:textAppearance="?attr/textAppearanceList
    <TextView
     android:id="@+id/listBondtermView"
     android:layout_width="wrap_content"
     android:layout_height="wrap_content"
     android:textAppearance="?attr/textAppearanceList
    <TextView
     android:id="@+id/listBondcouponView"
```

```
Microsoft Windows [Version 10.0.17134.1550]
(c) 2018 Microsoft Corporation. All rights reserved.
```

iOS versus Android

	Android	iOS/UIKit	iOS/SwiftUI
Screens	XML files with layout definitions	UIKit view, defined via Xcode	SwiftUI View struct
View Controller	Activity or Fragment class	UIViewController class	Combined with the screen
Navigation	Tabs or Activity/Fragment invocation via Intents	Screen transitions via segues	Tab and navigation views
UI/Model interaction	UI events trigger model changes; UI components query model, or asynchronously updated	As for Android	UI/Model bindings. UI is a function (Observer) of observable model data

Summary

- Manual coding of mobile apps too expensive & slow
- Replace manual coding by automated synthesis from platform-independent specifications
- Supports rapid evolution of apps
- Code-generators can be replaced/modified to support rapid evolution of mobile platforms/languages

- Tools: https://projects.eclipse.org/projects/modeling.agileuml

Planning for Industrial Practice needs to be Generic!

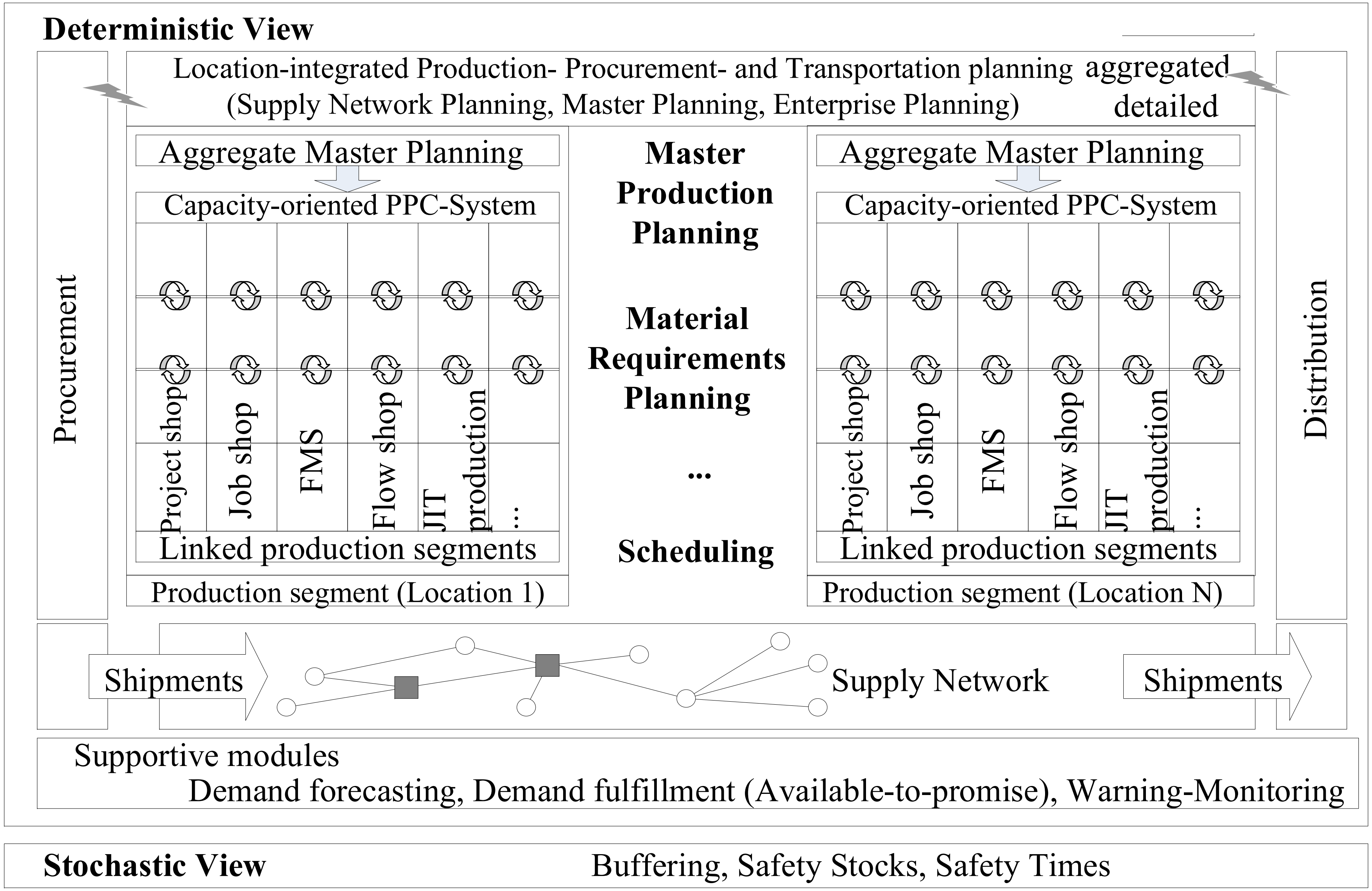
Professor Dr. Frank Herrmann
Innovation and Competence Centre for Production Logistics and Factory Planning (IPF)
University of Applied Sciences Regensburg
Postfach 120327, 93025 Regensburg, Germany
E-Mail: Frank.Herrmann@OTH-Regensburg.de

Panel Challenges in Complex Software Systems along with SoftNet 2020, Porto, Portugal
October 18 – 22, 2020

Professor Dr. Frank Herrmann – Curriculum Vitae

- born in Münster, Germany
- studied computer science at the RWTH Aachen University, Germany.
- at the Fraunhofer Institute IITB in Karlsruhe, Germany: worked primarily on algorithms for production control and received in this field a PhD.
- at SAP AG: several positions (Germany, Japan and USA), at the last as director.
- as a Professor for Production Logistics at the University of Applied Sciences in Regensburg, Germany: work mainly on planning algorithms, optimisation and simulation for operative production planning and control at companies.

Hierarchical Planning in ERP systems


Deterministic View
Location-integrated Production- Procurement- and Transportation planning
(Supply Network Planning, Master Planning, Enterprise Planning)
aggregated
detailed
Procurement
Distribution
Aggregate Master Planning
Capacity-oriented PPC-System
Project shop
Job shop
FMS
Flow shop
JIT production
...
Linked production segments
Production segment (Location 1)
Master Production Planning
Material Requirements Planning
...
Scheduling
Aggregate Master Planning
Capacity-oriented PPC-System
Project shop
Job shop
FMS
Flow shop
JIT production
...
Linked production segments
Production segment (Location N)
Shipments
Supply Network
Shipments
Supportive modules
Demand forecasting, Demand fulfillment (Available-to-promise), Warning-Monitoring
Stochastic View
Buffering, Safety Stocks, Safety Times

Real world application

Production System

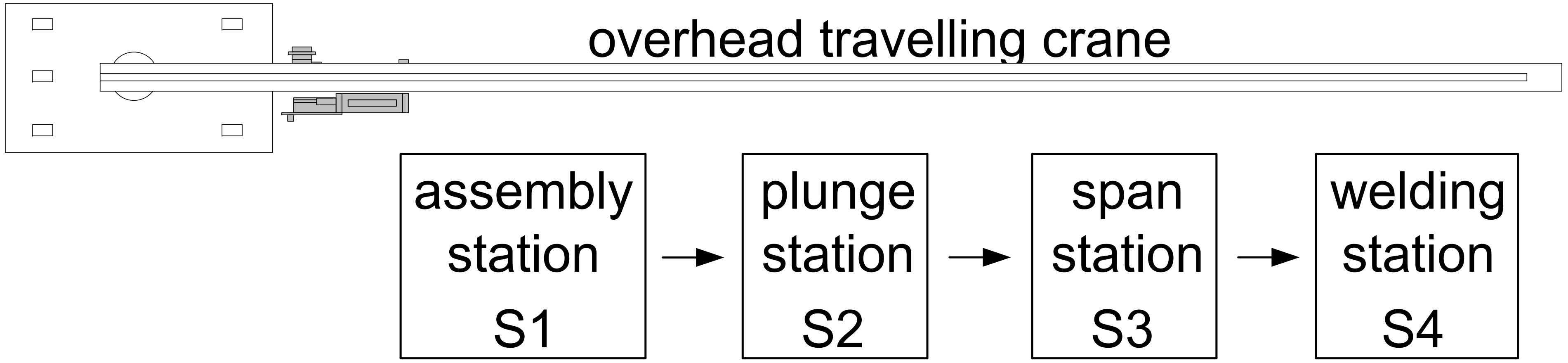
$$\begin{tikzcd} \boxed{S1} \arrow[r] & \boxed{S2} \arrow[r] & \boxed{S3} \arrow[r] & \boxed{S4} \end{tikzcd}$$

Recent years: highly specialized state-of-the-art solution algorithms.

- Manual transport system or automated guided vehicle.
- Limited Buffer between 2 stations - even no buffer or blocking.
- Performance criteria: tardiness, makespan, … .

Practitioner searches the literature for a suitable algorithm.

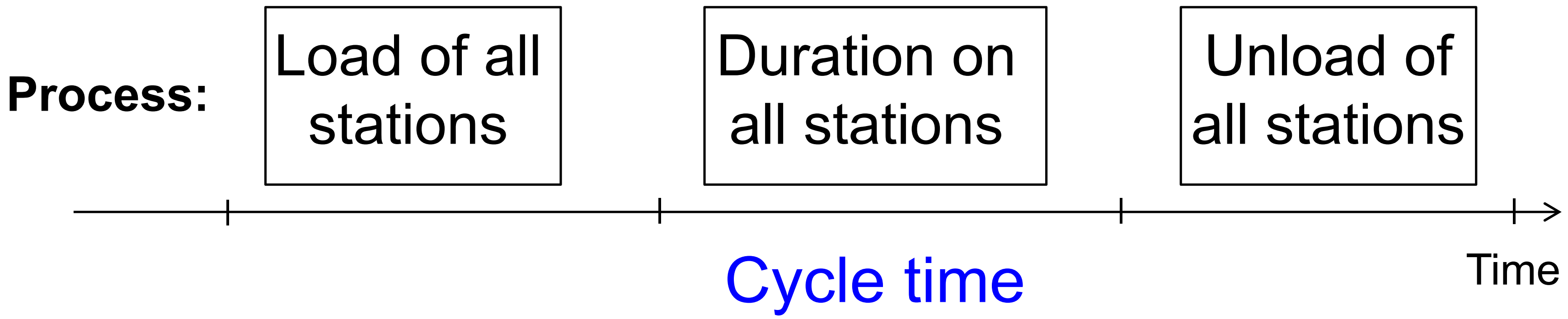
Real world application – as an example



Overhead travelling crane lifts a filter basket out of a station, transports it to the next station and **inserts it directly** in this station. This is just possible if this station is free.

- **No buffer** in the production line
- Feasible schedule of jobs is a **permutation** of these jobs.
- Other operational issues: **Move** of **crane** if all **stations** are **inactive**.
- No interruption of an operation: **transport after completion of all operations** - also for first and last operation. $\Rightarrow$ **“load”-restriction**
- A **station** may be **empty**.

Scheduling Problem 2/2



Main restrictions:

- This “load”-restriction.
- The no-buffer condition.
- The capacity of the stations.
- Relaxation of the “load”-restriction $\Rightarrow$ no-buffer problem which is **NP-hard in the strong sense** for more than two stations.

Further technolgical restrictions

Limited number of assembly ground plates.

Others are addressed in the special track “The Use of Simulation for Manufacturing Applications (SIMMaApp)” along with SIMUL 2020.

Consideration of

- Workers
- Maintenance
- Stochastics.

Proposal

Special designed heuristics ⇐ „No free lunch“

Approach models like Petri Nets see track SIMMaApp in SIMUL2020.

Linear Optimisation Models[1].

Benefit from ever improving implementations of the Simplex method.

[1] Baker K. Computational results for the flowshop tardiness problem. Computers & Industrial Engineering, 2013 (64), 812–816.

From Requirements to Implementation and Back

Radek Kočí

Brno University of Technology, Faculty of Information Technology
Czech Republic
koci@fit.vutbr.cz

https://www.fit.vut.cz/person/koci

BRNO UNIVERSITY OF TECHNOLOGY FACULTY OF INFORMATION TECHNOLOGY

ICSEA 2020, 18.-22.10.2020

Topic

Initial theses

- Complex set of changing requirements
- Very difficult to keep track of requirements and their implementation
- Changes in requirements or implementation have to be traceable and interconnected

Consequences

- Formal models for requirement specification and analysis
- Models and implementations are inextricably linked under the Simulation Driven Development

Formal methods

Formal specification

- predefined rules for determining the meaning of specifications
- written in formal languages
- supported by tools

$\Rightarrow$ enable rigorous software development

Formal description

- specifying requirements and desired properties
- modeling internal behavior
- the description is typically at certain level of abstraction
- precise, consistent and unambiguous

Formal methods

Formal specification

- formal specification let designers use abstractions and reducing the conceptual complexity of the system under development
- formal specification formalizes the statements describing element properties
- precise formulation of statements permits machine manipulation
- a more sophisticated form of validation and verification that can be automated using tools
- the specification may be mechanically transformed into another one, more detailed than its predecessor, and, eventually, into executable program

Simulation Driven Development

From Design to Target System

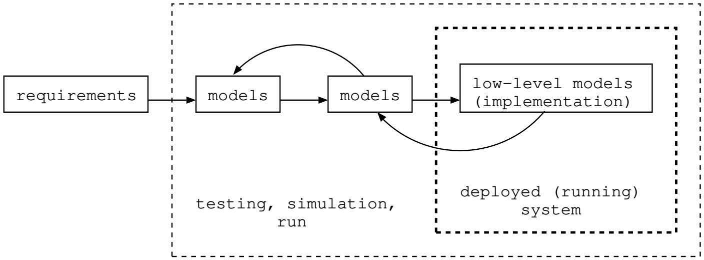
- the implementation is created by models transformations
- the resulted code has to often be changed by hand, there remains the problem of imprecision between models and implementation
- keeping changes over implementation and all used models is difficult

$$\begin{tikzcd}
\boxed{\texttt{requirements}} \arrow[r] & \boxed{\texttt{models}} \arrow[r] \arrow[l, bend left, "!"] & \boxed{\texttt{models}} \arrow[r] \arrow[l, bend left, "!"] & \boxed{\texttt{implementation}} \arrow[l, bend left, "!x"]
\end{tikzcd}$$

Simulation Driven Development

From Design to Target System and Back to Design

- shifts the point of software development from programming to modeling
- could be seen as model-based programming languages
- no (functional) need to transform models to the system implementation



Simulation Driven Development

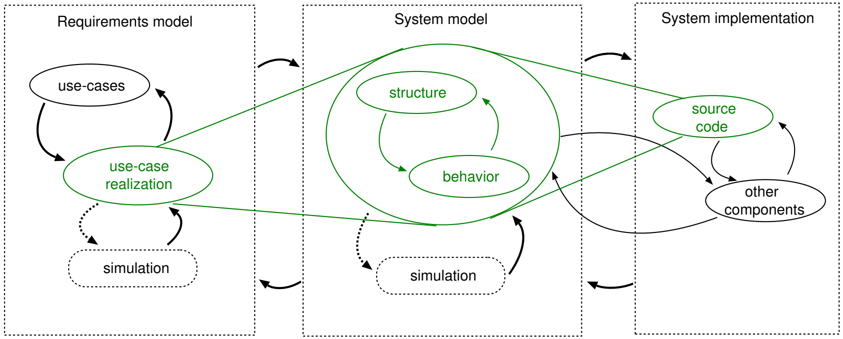
Principle

- Domain model – captures the concepts of the domain system as identified and understood
- Behavioral model – captures an external view of system functionality, its behavior, and interaction with the surroundings
    - User requirements modeling – use cases
    - Scenario modeling – behavior and interactions of individual cases
- Design model – sophisticated domain and behavioral models, more details
- Realization – implementation or deployed design models

Simulation Driven Development

Essential parts of the systems are presented through simulation (formal) models

- same or transferable models in design and implementation
- parts of the models may be supplemented by a code



Simulation Driven Development

Principle (cont.)

- Scenarios at the behavior level coincide with scenarios at the design level and are no longer distinguished.
- Continual development of behavior models becomes design models, which serve simultaneously for specification purposes.
- Design models can contain other objects from the domain environment to simulate the system or run under real-world conditions without having to show this implementation details at the requirements or behavior model.
- The same model can therefore be used for both documentation requirements and the executable version (prototype, implementation) of the developed system.

EXPERIMENTELLE VERBESSERUNG DES LERNENS VON SOFTWARE ENGINEERING

SOFTWARE ENGINEERING EDUCATION MATTERS!

Dieter Landes & Yvonne Sedelmaier

dieter.landes@hs-coburg.de, yvonne.sedelmaier@hs-coburg.de

Supported by
Bundesministerium für Bildung und Forschung (Grants 01PL12022A / 01PL17022A)

Presenters‘ resume

- **Dieter Landes** is a full professor of software engineering and database systems at the University of Applied Sciences, Coburg, Germany. His research interests are in requirements engineering, software engineering education, and data mining.

- **Yvonne Sedelmaier** holds a diploma and a PhD in pedagogy with a major focus on adult learning and continuing education at the University of Bamberg, Germany. Her research interests are teaching and learning software engineering at universities and software engineering didactics.

Motivation (1)

- Complexity of (software) systems increases in many aspects
  - New types of applications
    - E.g. adaptive…
  - New groups of stakeholders
    - E.g. elderly people
  - New technologies
    - E.g. Artificial Intelligence …
  - Fundamentally transdisciplinary

Motivation (2)

- Building complex systems successfully presupposes more and better (software) engineers and developers
- Challenges for
  - Universities
  - Companies
- What and How???

Competence-Orientation (1)

- Knowing new technologies is fine (but still hard)
- Successful development also requires many skills, i.e. non-technical competences
  - Cope with complexity
  - Work with other disciplines within the team and beyond
  - Critical thinking
  - Creativity
  - Learn to learn continuously

Competence-Orientation (2)

- Competences (technical and non-technical) constitute educational goals
  - Cannot be taught
  - Can be exercised
  - Individual needs differ
- Not all required competences can be addressed in a single module
  - Specific focus
  - Varying scope

New Learning Concepts Required (1)

- Learner-centered, competence-oriented approaches
  - Less instruction
  - More activating, self-directed learning
  - Challenge in SE education
    - Real-world complexity cannot be mirrored in university settings right away
    - But: real-world complexity required
      - Programming is not software engineering in the large

New Learning Concepts Required (2)

- Competences First
  - Imply intended learning outcomes (WHAT?)
    - Settings follow,
    - Contents follow,
    - Learning technologies follow,
    - Examination styles follow
    - as means to an end (HOW?)

Software Engineering Education as a Transdisciplinary Challenge

- Software Engineering experts are experts
  - for software engineering issues,
  - but not necessarily for education

- Joint endeavor of software engineers and experts for adult learning / higher education / didactics (pedagogy, andragogy)

EVELIN

EXPERIMENTELLE VERBESSERUNG
DES LERNENS VON SOFTWARE ENGINEERING

Questions?

Thank you for your attention!

PROJEKT EVELIN
Experimentelle Verbesserung
des Lernes von Software Engineering

GEFÖRDERT VOM

Bundesministerium
für Bildung
und Forschung