Component-Based Software Engineering (CBSE) Announcements

Prof. Dr. Uwe Aßmann
Technische Universität Dresden
Institut für Software- und Multimediatechnik
http://st.inf.tu-dresden.de
13-0.3, 18.04.13

Elements of the Course

- Lecturing
  - Do not miss one, they should give you a short and concise overview of the material
- Reading
  - Slides on “Obligatory Literature” require you to read papers from the web
    - TU Dresden has subscription to ACM Digital Library and IEEE Explorer
  - Slides on “Secondary Literature” contain useful but optional literature
- Exercise with Christoff Bürger
  - Exercise sheets
    - Handed out every week, with some breaks
    - You have one week to solve them on your own
    - After that, solutions will be explained in the Exercise
- Meeting hour Thursday, 11:00-13:00
  - Book a slot with Katrin Heber, 0351 463-38463
  - Katrin.Heber@tu-dresden.de
- Oral exams (20 min) usually in September, so that you have enough time to learn

Reading Along the Lectures

- Unfortunately, the course is not covered by any book
  - About 60% is covered by the blue book “Invasive Software Composition”
  - Most of the rest on classical component systems by Szyperski in the book “Component Software. Beyond object-oriented computing. Addison-Wesley.”
- You have to read several research papers, available on the internet
  - Marked by “Obligatory Literature”
- Secondary Literature is non-mandatory, but interesting reading. Can be done during the course
- Other Literature is not to be read, but also interesting.

Obligatory Literature

- During the course, read the following papers, if possible, in sequential order.
- Every week, read about 1 paper (3-4h work)
- Course web site

Obligatory Literature

- [ISC] U. Aßmann. Invasive Software Composition. Springer, 2003.
- C. Szyperski. Component software. Beyond object-oriented computing. Addison-Wesley. Bestseller on classical component systems.

Papers

- [McIlroy68] D. McIlroy. Mass-produced Software Components. 1st NATO Conference on Software Engineering.
- [Dami95] Laurent Dami. Functions, Records andCompatibility in the Lambda N Calculus in Chapter 6 of “Object-oriented Software Composition”. http://scg.unibe.ch/archive/oosc/PDF/Dami95aLambdaN.pdf
- CORBA. Communications of the ACM, Oct. 1998. All articles. Overview on CORBA 3.0.
- Others will be announced.

Recommended Literature

- Oscar Nierstrasz, Dennis Tsichritzis. Object-oriented Software Composition. Web book. http://scg.unibe.ch/archive/oosc/download.html
- I. Forman, S. Danforth. Meta-objects in SOM-C++. Very good book on meta object protocols and meta object composition.
- Journal Software - Tools and Techniques. Special Edition on Componentware, 1998. Springer. Good overviews.
- R. Orfali, D. Harkey: Client/Server programming with Java and Corba. Wiley&Sons. Easy to read.
- CORBA. Communications of the ACM, Oct. 1998. All Articles.

Recommended Literature

- [GOF, Gamma] E. Gamma, R. Helm, R. Johnson, J. Vlissides: Design Patterns. Addison-Wesley 1995. Standard book belonging to the shelf of every software engineer.
  - The book is called GOF (Gang of Four), due to the 4 authors
- Alternatively to GOF can be read: [Remark: If you have already studied GOF intensively, do not read these]
  - A. Tesanovic. What is a pattern? Paper in Design Pattern seminar, IDA, 2001. Available at home page.
  - On Composite, Visitor: T. Panas. Design Patterns, A Quick Introduction. Paper in Design Pattern seminar, IDA, 2001. Available at home page.
  - P. Pop. Creational Patterns. Paper in Design Pattern seminar, IDA, 2001. Available at home page.

Less Important

- K. Czarnecki, U. Eisenecker. Generative programming . Addison-Wesley 2000. Good overview on aspects, but not on components
- F. Griffel. Componentware. dpunkt-Verlag. In German. A lot of material.

Please, Please Be Aware – There Will Be Pain!

- This course is not like a standard course
- It treats rather advanced material, the concept of graybox engineering
- No single book exists on all of that at all
  - ISC covers about 60%
  - Please, collaborate!
  - Read the articles
  - Ask questions!
  - Do the exercise sheets
- The exam can only be done if you have visited all lectures and solved all exercise sheets
- Learn continuously! One week before the exam is too late!
- Be aware: you have not yet seen larger systems
  - Middle-size systems start over 100KLOC

The Positive Side

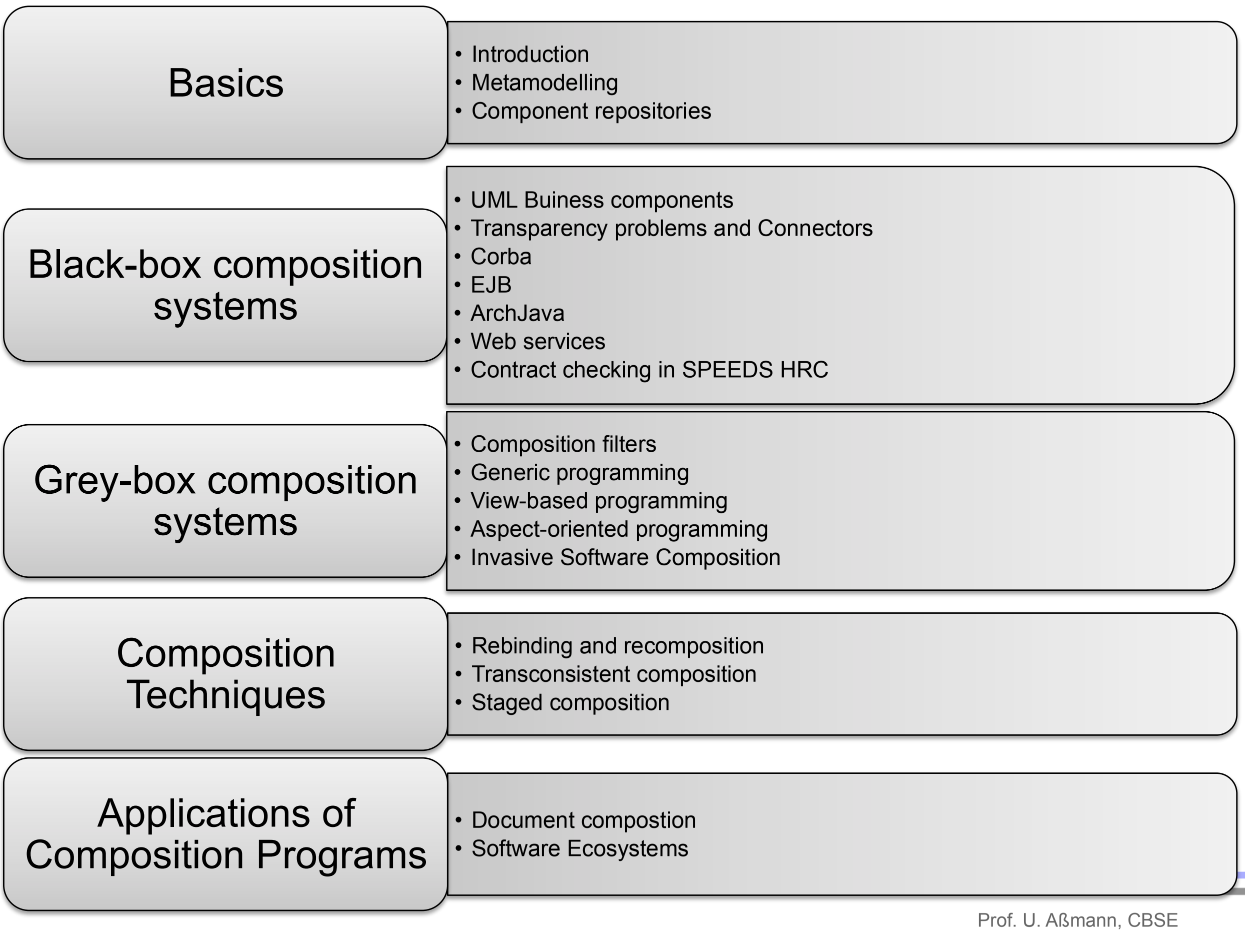
- If you follow carefully, you will discover an exciting world of graybox composition, a new way to *extend* software
- The gain is worthwhile the pain!

Component-based Software Contents and Goals

CBSE, © Prof. Uwe Aßmann

Course Contents

- **Part 0: Basics**
  - History and overview: Criteria for composition
  - Basics: Reflection and metaprogramming, Meta-object protocols (MOP), Metadata,
  - Finding components with faceted metadata and protocol conformance
- **Part Ia: Classical component systems (Simple black-box composition systems)**
  - Business components
  - Classical component systems: Development Process, Problems
  - Enterprise Java Beans (EJB)
  - Quality-controlled compositiion systems (QCS)
- **Part Ib: Architecture systems and languages (Advanced black-box composition systems)**
  - Corba
  - Web services
  - Architecture Systems
- **Part II: Gray-box composition systems (Invasive composition)**
  - Calculi for component systems
  - Composition Filters
  - Generic Programming (BETA)
  - View-based programming: Hyperspace programming
  - Aspect-oriented software development: AOSD and AOP
  - Invasive software composition
- **Part III: Composition techniques**
  - Uniform composition based on metamodels
  - Rebinding and recomposition
  - Transconsistent composition
  - Staged composition
- **Part IV: Applications of composition systems**
  - Universal Composition
  - Invasive Model Composition
  - Transconsistent document composition



Basics

- Introduction
- Metamodelling
- Component repositories

Black-box composition systems

- UML Buiness components
- Transparency problems and Connectors
- Corba
- EJB
- ArchJava
- Web services
- Contract checking in SPEEDS HRC

Grey-box composition systems

- Composition filters
- Generic programming
- View-based programming
- Aspect-oriented programming
- Invasive Software Composition

Composition Techniques

- Rebinding and recomposition
- Transconsistent composition
- Staged composition

Applications of Composition Programs

- Document compostion
- Software Ecosystems

Component-Based Software Goals

Main Goals

- Understand the concept of a *component model*
  - Frameworks and product lines work with various different component models
  - Variability, extensibility, and glueing are three central goals
  - There are other central concepts for component models than classes and objects
- Understand *composition systems*
  - Understand grey-box, fragment-based composition
  - why it introduces new forms of static extensibility
  - why other static component models are special cases of it
- Understand different times of composition
  - dynamic composition
- Understand components as collections of standardized role types
- Understand connectors as role models plus protocol

The End