Lectures and Seminars
(Vorträge und Seminare)
Offers/Angebote ¹ (status: August 20, 2015)

Table of Contents

1 Lectures/Vorträge
  1.1 Basic Architectural Decisions in System Design ........................................ 2
  1.2 Technological/Sociological Synergies in ICT ............................................. 2
  1.3 Adapting Software Products to Cultural Differences (Localization) .............. 3
  1.4 System Development Process Technology ..................................................... 3
  1.5 Quality and Maturity of System Development Processes .............................. 4
  1.6 Psychological Barriers in Technology Transfer ............................................ 4
  1.7 Systemic Aspects of Disaster Responses .................................................... 4
  1.8 Motivational Issues in Component Based Development ............................... 5
  1.9 e-teaching ..................................................................................................... 5
  1.10 Software Inspections .................................................................................... 5
  1.11 Software Archaeology .................................................................................. 5
  1.12 Faust - der erste Programmierer (only for German-speaking audiences!) ........ 6

2 Seminar (englisch): Human Factors in Systems Engineering .......................... 6

3 Seminar (deutsch): Menschliche Faktoren im Systems Engineering ................ 7

4 Curriculum Vitae - em. o.Univ.-Prof. Dipl.-Ing. Dr. Gerhard Chroust, M.S. ...... 8

5 References ......................................................................................................... 8

¹www.gerhard-chroust.at/gc-listings/gc-angebote.pdf
1 Lectures/Vorträge

I can offer to hold the lectures listed below

General Remarks:
- Language: German or English.
- Duration: the standard duration is 20 to 45 minutes, depending on desired depth and interest of audience.
- All lectures will be accompanied by power-point slides in the language of the lecture.
- All lectures are based on conference/journal contributions (see section 5), but almost exclusively in English.
- All lectures were already held once or several times at conferences or meetings (see section 5).
- The given abstracts were taken from of the cited papers.

Bemerkungen:
- Sprache: Deutsch oder Englisch.
- Dauer: die übliche Dauer ist zwischen 20 und 45 Minuten, je nach gewünschter Tiefe und nach Interessenlage des Publikums.
- Die Vorträge werden von Power-point-Folien in der Vortragssprache begleitet.
- Alle Vorträge basieren auf Publikationen auf Konferenzen oder in Fachzeitschriften (siehe Abschnitt 5), aber fast ausschließlich nur in Englisch.
- Alle Vorträge wurden ein oder mehrmals im Rahmen von Konferenzen oder Tagungen gehalten (siehe Abschnitt 5).
- Die Kurzfassungen wurden meist einer der zitierten Veröffentlichungen entnommen.

1.1 Basic Architectural Decisions in System Design

References: [Chroust-05a] [Chroust-05t] [Chroust-07b] [Chroust-08a] [Chroust-10h] [Chroust-10q] [Chroust-12z]

Abstract: The advent of the computer has changed our industrial, commercial and social live with a tremendous speed and profoundness. As a consequence the complexity of system to be designed has grown, too. When designing a complex software-intensive system it is unavoidable to make some a-priori basic assumptions about its architecture. We introduce so-called Basic Architectural Alternatives as a means to guide these decisions and to understand their effects. These alternatives are classified according to five fundamental dimensions (enactment time, location, granularity, control, and automation and task distribution). For each dimension we describe several typical, real examples of the alternative. For each example we indicate fundamental properties of that alternative: elasticity with respect to later compromises, reversibility of the choice (with reasonable effort), uniformity requirements with respect to the different elements of the system, and applicability of the chosen method across the dimension. Finally we discuss synergetic or contradictive effects of alternatives with respect to others. We hope that this analysis together with the specific examples and their key properties provide some insight for novices and seasoned designers and guides them during the early phases of system design.

Keywords: system architecture, architectural trade-offs, system properties, design decisions, dichotomic alternatives, system types, wicked systems

Characteristics: Only for technical audiences with some (not necessarily deep) knowledge in programming and related software areas.

1.2 Technological/Sociological Synergies in ICT

References: [Chroust-04u] [Chroust-05m] [Chroust-05p] [Chroust-05ac] [Chroust-08a] [Chroust-08k] [Chroust-08za] [Chroust-11f] [Chroust-14h]

Abstract: The Information and Communication Technology (ICT) has thoroughly changed our private and public life and the structure of our society with enormous speed. On the technological level we observe a strong positive feedback between several technological innovations (e.g. faster computer hardware allows to run more
sophisticated software which is used to design even faster computer hardware). Much stronger seems the feedback between the technological capability of computers and human activities in business and society. Here computers have not only accelerated existing processes, they have caused dramatic paradigm changes in areas like e-business, e-learning, information management, communication patterns etc. preparing and forming the so-called "information society" or "knowledge society".

We discuss technological innovations, the changes they imply, and the resulting impacts. Then we discuss paradigmatic changes for individuals, business, society, and technology, some of which have a strong feedback on new technological approaches.

Characteristics:

Long version: Only for technical audiences with a good (not necessarily deep) knowledge in programming and related software areas.

Short version: not very technical, future oriented, based on a systemic view, slightly philosophical powerpoint-presentation:

1.3 Adapting Software Products to Cultural Differences (Localization)

References: [Chroust-07d] [Chroust-08b] [Chroust-08h] [Chroust-08j] [Chroust-09zg] [Chroust-11k] [Chroust-12zd] [Chroust-12ze] [Chroust-15d]

Abstract: The pervasiveness of the Information and Communications Technologies (ICT) results in a tremendous amount of software products being offered, largely global electronic market using Internet. Software providers have to market and sell software products in many different countries, bringing many more people into direct (often unexpected) contact with computerized interfaces. Potential customers expect the software product not only 'to speak their language' but also to show a behavioral pattern which is compatible with their cultural expectations and preconditions. They expect reactions from software product like an "intuitive, courteous butler". This involves a great deal more than a pure language translation: it implies the transfer of the software product into another culture taking into account all aspects of cultural divergence. We speak of localization.

With respect to the necessary adaptations we identify seven layers of localization of increasing cultural dependency and sensitivity which range from 'Technological Infrastructure', e.g. providing for the correct coding for special national characters, to the 'Cultural Layer' catering for highly complex cultural traditions and expectations like social ranking and taboos. The lecture is based on research by Hampden and Trompenaars’s six Cultural Dimensions [Hampden-00] and on Hofstede and Hofstede’s five Cultural Dispositions [Hofstede-05] and is supported by extensive examples. We also give Some ideas on remedies for cultural mismatches and insensibility close the paper.

Keywords: localization, cultural adaptations, user interface, national differences, conventions cultural dimensions, cultural dispositions

Characteristics:

very important subject with respect to globalization. Not difficult to understand also for non-technical audiences. Examples of mishaps provide some hilarious moments.

1.4 System Development Process Technology

References: [Chroust-92a] [Chroust-96h] [Chroust-00f] [Chroust-08h] [Chroust-08j] [Chroust-10v] [Chroust-11e] [Chroust-12y] [Chroust-13h] [Chroust-14a]

Abstract: A key to a successful systems engineering project is an orderly development process to conceptualize, design, build, and evaluate the intended system. This process has to be formally defined by a Process Model, enacted and its observation often even enforced. This includes the need to document the defined process with an appropriate modelling language. Obviously all project members have to follow the ‘same’ process. In “classical industries” such processes have been established since long (architecture: several thousand years, car industry: 150 years). The arrival of the ICT (Information and Communication Technologies) has added new perspectives, new challenges, and new methods [Chroust-96h, Chroust-10v], especially with respect to industries which traditionally relied on electro-mechanical implementations. The notion and basic concepts of process view are presented together with alternatives for describing and enacting a process model. Human aspects are discussed in some detail. Other disciplines gradually adopt process thinking (e.g. Disaster Management [Haider-14q]).

Keywords: System Engineering, Process Model, process view, process thinking, modeling, enactment

Charakteristik: moderately technical, most concepts are applicable to a very wide range of technical developments.
1.5 Quality and Maturity of System Development Processes

**References**: [Chroust-94c] [Chroust-95w] [Chroust-96x] [Chroust-97g] [Chroust-98y]

**Abstract**: After a short discussion of the notions of software development process and its relation to the quality of software products, we discuss several approaches to measuring the quality of a software development process (its "Maturity"). We present relevant assessment methods like CMM, Bootstrap, ISO 9000, ISO/IEC 15504 and ISO/IEC 30000-family. Especially ISO/IEC 15504 is treated in more detail. Several economic questions are discussed relating to the achievement of certain capability levels and the return on investment for that effort.

**Keywords**: Process Model, enactment, maturity, quality assessment, assessment methods, certification

**Charakteristik**: rather technical, discussing quality standards like ISO 9000, ISO/IEC, and ISO/IEC30000. A good understanding of software development processes is prerequisite.

1.6 Psychological Barriers in Technology Transfer

**References**: [Chroust-02d] [Chroust-08j] [Chroust-09h]

**Abstract**: The introduction of computer-supported process models for the development of software ("software engineering environments"). arouses often considerable distrust and resistance. In this lecture we identify, analyze and discuss psychological/motivational fears, resistance and obstacles to the acceptance of such modern environments. Linking those fears to the 5 Levels of the Maslow Pyramid of Needs allows to structure the various obstacles and approaching them individually. The lectures is based on personal experience with the introduction of IBM’s software engineering environment ADPS [Chroust-88g, Chroust-89d] and with the EU-projects ESPITI [Chroust-96a] and SPIRE [Chroust-97i, Chroust-98m].

**Keywords**: resistance, fear, Maslow Pyramid, satisfaction, process models, distrust

**Characteristics**: In many cases the introduction of even useful and user-friendly software fails due to often deep seated emotional barriers. We try to identify the underlying reasons by mapping the users’ desires and fears by mapping the emotions against the Maslow Pyramid of Need.

1.7 Systemic Aspects of Disaster Responses

**References**: [Chroust-08v] [Chroust-08zc] [Chroust-09j] [Chroust-09m] [Chroust-09v] [Sturm-09s] [Chroust-10c] [Chroust-10f] [Sturm-10n] [Chroust-10f] [Chroust-11e] [Chroust-11g] [Chroust-11j] [Chroust-12c] [Chroust-15b] [Chroust-12s] [Chroust-12d] [Chroust-13g] [Chroust-14a] [Chroust-15b] [Chroust-12s] [Chroust-12d] [Chroust-13g] [Chroust-14a]

**Abstract**: Today’s catastrophes (many of them man-made or at least triggered by human activities) frequently endanger a growing number of humans and larger areas in more diversified ways, creating a need for dependability and resilience of our environment.

Experience tells us that no matter what precautions and quality approaches we take we will always encounter systems which initially were dependable and ‘suddenly’ turn untrustworthy due to some unexpected, unknown cause. A system which in itself is unable to reestablish its dependability, i.e. it is not resilient (any more) needs an outside intervention: For humans a physician acts as an intervention system for re-establishing dependability. A complex system can be made resilient by the inclusion of an Intervention System which intervenes in the case of loss of dependability.

We investigate the role of First Responders (i.e. fire brigade, ambulance services, police forces) as an Intervention System, aimed at providing resilience. We identify properties of different catastrophes and their implications for the activities of First Responders both in training and actual interventions by simulation of events, Mixed Reality tools, etc.

**Keywords**: Catastrophe, intervention, First Responder, systems view, compensation, human aspects CBRN-emergencies, process modelling, resilience, dependability, simulation, Mixed Reality

**Characteristics**: Special attention is paid to the training and the behavior of First Responders and to the needs of CBRN (chemical, biological, radioactive, or nuclear causes) incidents. We discuss the various behavior types of disasters and discuss interventions both from a human point of view and seen as processes.
1.8 Motivational Issues in Component Based Development

References: [Chroust-03g] [Chroust-04a] [Chroust-06a]

Abstract: Despite all progress in technology and considerable publication effort, component based software development (CBD) seems to get a slower start than anticipated. Some of the reasons seem to be caused by soft factors (psychological and motivational) of software developers. We describe, based on Maslow’s Hierarchy of Needs, these factors, cluster them into three major areas and discuss some countermeasures.

Keywords: Component Based Development, CBD, Maslow, psychological barriers, motivation, software developers

Characteristics: Not difficult to understand also for non-technical audiences since the main focus is the motivation of engineers. Problems and issues of adoption are explained.

1.9 e-teaching

References: [Chroust-03b] [Chroust-05u] [Chroust-05v] [Chroust-06e] [Chroust-07e] [Chroust-08m] [Chroust-08n] [Chroust-12v]

Abstract: We review how changing paradigms of the Information and Communication technologies affect essential parameters of creating and disseminating information in the academic world. Using a historical perspective their impact on the university education system will be discussed. Both the effects of changing types of documents (e.g. electronic documents) and communication means (e.g. Internet) are investigated. The parameters are analyzed with respect to their effects on the timing of the various subprocesses of academic education (e.g. time to publication, time to teaching, etc.). Consequences for quality attributes and teacher/student relationships are discussed in view of the academic education processes. We show that many of the cherished traditions, habits and beliefs of yesterday are invalidated by the modern IC-technologies.

Keywords: Educational Processes, Information and Communication Technologies, IC-technologies, academic education, time delays

Characteristics: not very technical, discusses important issues of e-teaching from the teacher’s perspective. Some not-so-obvious conclusions are drawn.

1.10 Software Inspections

References: [Chroust-99g] [Chroust-05e]

Abstract: In 1976 Mike Fagan published his seminal paper on inspections of software development documents based on IBM’s field experiment data. He suggested not to wait with validation and verification of a software product until running code was available but introduce intermediate so-called “inspections” after every major milestone of a project.

Despite all documented advantages inspections did not receive the wide-spread use they deserved. A major reason are psychological/motivational problems and fears of the developers.

After introducing the concepts of inspections we discuss general problems with inspections, especially those related to motivational/psychological causes. We close with some extensions/modifications of the original concept utilizing modern means of ICT.

Characteristics: Not too difficult to follow, advocates a very important technique for quality improvement. Discusses pros and cons.

1.11 Software Archaeology

References: [Chroust-04e] [Chroust-05b]
Abstract (english): Maintenance is one of the key problems of software engineering, often nicknamed ‘software archaeology’. But maintenance is also closely related to software reuse, one of the keys to productivity. We discuss analogies between software maintenance/software-reuse and archaeology, emphasizing similarities and dissimilarities. It shows some surprising parallels and insights concerning what one calls legacy systems or legacy artifacts.

We discuss the so-called "RE"-techniques (reverse engineering, repair, redocumentation, re-location, ...) and discuss them in terms of archeology and of software. For the archeology part pictures of historical sights and artefacts of either Assyria or Vienna are provided (depending on the audience). The visual artefacts of the past give sometimes a visual analogy to the problems of software maintenance.

Characteristics (english): Even if the topic is to be taken with a grain of salt, the lecture conveys important messages and problems of maintenance and ‘visualizes’ them with pictures from archaeological sites. Well received from technical and mixed audiences. The special charm is that I show lots of pictures of the archaeology of Vienna.


Charakterisierung (deutsch): Wenn auch die Aussagen des Vortrages mit etwas Vorsicht zu verstehen sind, so beschreibt er doch wesentliche Fragen und Einschränkungen sowohl der Software-Wartung als auch der Archäologie durch die Bildunterstützung. Der Vortrag wird sowohl von technisch interessierten als auch von allgemeinen Zuhörern immer gut aufgenommen. Durch die Bilder von Vindobona und Karnuntum (die römische Stadt nahe bei Wien) ist er auch für Wien-Besucher interessant.

1.12 Faust - der erste Programmierer (only for German-speaking audiences!)


Characteristics:
Obwohl nicht viel Tiefgang, kann der Vortrag ein Publikum, besonders wenn es etwas Theater-Liebe hat, sehr erheitern.
Kann nur auf Deutsch gebracht werden.

2 Seminar (englisch): Human Factors in Systems Engineering

References: [Chroust-02d] [Chroust-13g]

Language: English

Duration: the standard duration is 8 to 15 hours, depending on desired depth and interest of audience.

Sources: The material combines and puts into perspective several of the topics individually discussed in the lectures, see section 1. The contents is largely identical to the German version (see section 3).

Presentation: The seminar will be presented via power-point slides in the language of the lecture.

Handout: The lecture is accompanied by handouts in the language of the lecture, containing all slides plus chapter headings, reference lists, and some additional material.
Experience: The seminar was held several times at the Johannes Kepler University Linz as part of the informatics-curriculum (in German) and 4 times in English (1x Sri Lanka, 3x Hungary).

Contents: Systems Engineering is largely a human-centered activity. Its success depends mostly on human involvement, ingenuity, motivation and team work. Systems are designed by humans and are provided with interfaces which again provide communications with humans. This seminar focuses on human beings and their behavior in relation to the development and usage of software-intensive systems. Both aspects, development and usage, by necessity are subject to sociological and cultural influences. The focus is on two issues: General Human Cooperative Behavior and Cultural Differences. This is applied both to the development processes for software and to the appearance of the produced software itself. The course will create a basic understanding of the issues involved in order to make both systems development and system usage more human oriented.

Structure:

- Part 1: Basic Human Behavior: This provides the basis: human behavior - the individual, Transactional Analysis, groups, group dynamics and teams, human needs, Maslow pyramid, motivation, creativity, computers as capability enhancers, social computing
- Part 3: Human Factors of system development processes: Social Competence, Self-understanding and Ethnic of developers, Project Management, the project leader, Evaluations and Criticizing

Characteristics:
Mostly oriented towards human factors, human problems and issues as related to technical and managerial problems in a multicultural world (e.g. outsourcing, international teams), including psychological aspects of humans.

3 Seminar (deutsch): Menschliche Faktoren im Systems Engineering

Sprache: Deutsch

Dauer: die übliche Dauer ist zwischen 8 und 15 Stunden, je nach gewünschter Tiefe und nach Interessenslage des Publikums.

Quellen: Das Material beruht auf mehreren der Themen, die in den individuellen Vorträgen angesprochen werden (). Der Inhalt ist größtenteils ident mit der englischen Seminarversion (siehe Abschnitt 2)

Präsentationsform: Die Vorträge werden von Power-point-Folien in der Vortragssprache begleitet.


Erfahrung: Das Seminar wurde mehrmals an der Johannes Kepler Universität im Rahmen des Informatik-Studienplans (in Deutsch) gehalten und 4 mal in Englisch (1x Sri Lanka, 3x Ungarn).


Struktur:
• Teil 1: Die Basis: Menschliches Verhalten: Menschliches Verhalten, das Individuum, Transactions-Analyse, Gruppen, Gruppendynamik und Teams, menschliche Bedürfnisse, Maslow-Pyramide, Motivation, Kreativität, Computer als Fähigkeitsverstärker, Social Computing
• Teil 2: Kulturelle Differenzen (nach Hofstede & Hofstede und Hampden & Trompenaars); Dimensionen kultureller Unterschiede, internationale Systementwicklung, menschliche Probleme bei Outsourcing, Lokalisierung und Internationalisierung von Software-Produkten
• Teil 3: Menschliche Faktoren in Systementwicklungsprozesse: Soziale Kompetenz, Selbsterständnis und Ethik von Entwicklern, Projektmanagement, der Projektleiter, Evaluierung und Kritik


4 Curriculum Vitae - em. o. Univ.-Prof. Dipl.-Ing. Dr. Gerhard Chroust, M.S.

Gerhard Chroust joined the IBM Laboratory Vienna in 1966 and co-developed until 1968 the Formal Definition of PL/I, followed 1969-72 by research on compiler construction. 1972 to 1976 he was assistant to the Laboratory Director, Heinz Zemanek, handling the cooperation with academic institutions. 1976-82 he jointly developed the PL/I Compiler for the IBM 8100. 1983-90 he was member of the development team for ADPS (Application Development Project Support), responsible for defining the Process Model. 1990-91 he acted as a product support representative for ADPS. From 1992 to 2007 he was full Professor of Systems Engineering at the Kepler University Linz, Austria initially at the Institute for Systems Sciences, and later Head of the Institute of Systems Engineering and Automation. Since 2007 he is professor emeritus.

Current research and teaching interests are focussed on cultural differences and human factors in system development, applications of systemic thinking, description, implementation and enactment of socio-technical systems and the support of the necessary development processes. Further research is devoted to disaster management.

Dr. Chroust holds a Diplom-Ingenieur and a PhD. from the Technical University of Vienna and a M.S. from the University of Pennsylvania.


He is Secretary General of the International Federation of Systems Research (IFSR), a former president of the Austrian Society of Informatics (ÖGI) and a former vice-president of the Austrian Society for Cybernetic Studies.

He is the Editor-in-Chief of the Books Series ‘Informatics’ (Trauner Verlag Linz), Editor-in-Chief of the IFSR Newsletter, and a former head of the Editorial Board for the Book series of the Austrian Computer Society.

Contact:
Univ.-Prof. Dr. Gerhard Chroust, M.S.
Donaustr. 101/6
A-2346 Maria Enzersdorf
Austria
Tel.: (mobile) + 43 664 28 29 978
email: Gerhard.Chroust@jku.at
www: http://www.chroust-gerhard.at

5 References

Literatur

[Chroust-00f] CHROUST, G. Software Process Models: Structure and Challenges in: Feng, Y. and Notkin, D. and Gaudel,


