RONALD FISCHER

Häberlstraße12, 80337 München
Phone: 089-532265
Mobile: 0171-5074874
E-Mail: ronald.fischer@fusshuhn.de

This profile was updated: ***11 August 2014***
An ***up-to-date*** *version of this profile in* ***German*** *can be found at:* ***http://www.fusshuhn.de/business.htm***

OVERVIEW

Born:	15. June 1956 in Graz (Austria)
Languages:	German (native Speaker), English (fluent, sufficient for technical discussion), Portuguese (average, sufficient for smalltalk), Japanese (average, sufficient for small talk), French (learned at school), Greek (very basic knowledge only)
Industrial sectors I've worked in:	Telecommunication, Semi-Conductor Industry, Security Information Systems, Airplane Manufacturing, Insurance, etc.
Developing software since:	1977 (until 1980 in parallel to studying at the university)
Experience in the following software related topics:	Softwaredesign and -development Rapid Prototyping Object-oriented Programming Testautomatization & Regression Testing Designing Testplans Quality assurance (Test metrics, Code reviews, Coding guidelines etc.) Automated Software Deployment Developing software for embedded systems Agile Development (Scrum, Kanban)
The majority of my projects was focused on:	Implementation of complex software systems using RAD-languages, such as Ruby, Perl or Python, and Shell Scripts. Experience in developing security-relevant, mission-critical applications. Design and development of testing frameworks, testing plans and software for regression testing and quality control. Development of large systems in C++ under Linux/Unix.

Education:

Title:	Diplom-Ingenieur, , i.e. graduate engineer at "Technische Universität Graz" in Austria
Graduation year:	1980
Field of study:	"Technischen Mathematik mit Schwerpunkt Informations- und Datenverarbeitung", i.e. mathematics with focus on information technology

Knowledge

(+ = Basic knowledge, ++ = Good knowledge, +++ = Deep knowledge)

Software Development

Programming languages:	C (+++) C++ (+++) Perl (+++) Python and Jython (++) Ruby and JRuby (+++) Shell Scripting [bash,zsh] (+++) APL (+) Assembler [Z80,8086] (++) Assembler [Siemens System 300 Macro Assembler] (++) awk (++) FORTRAN 77 (++) SQL (++) Tcl (++) Haskell (+) Lisp (emacs lisp) (+) Java (+) NIAL (+) Pascal (++) REXX (+)
Web-related technologies:	JavaScript (++) HTML/XHTML (++) CSS (++) SCSS (++) HTML5 Canvas (++) Ruby on Rails (++) HTTP/REST (+) Markdown (+) jQuery/DOM (+) Java Server Pages (+) PHP (+) osCommerce (+)
Databases:	MySql (++) Oracle (++) Informix (+) JDBC (+) SQLite (+)
APIs/Libraries/Frameworks:	Expect [Tcl/Expect] (++) Tk [Tcl/Tk, Ruby/Tk, Python/tkinter] (++) Log::Log4Perl (++) pthreads (+)

Application software:	LSF [Load Sharing Facility, by IBM Platform Computing] (++) OpenOffice/NeoOffice (+)
Other:	XML (+++) XML Schema (+++) Clearcase (++) Subversion (++) Make and Clearmake (++) Emacs and jEdit (++) M4 (++) PuTTY and ssh (++) ftp/sftp (++) git (+) ASN1 (+) Ant (+) UTP, Bugzilla and ClearQuest (+) gdb (+)
Testing:	Development of logging/tracing tools (+++) Integration Test (+++) Unit Test (++) Testautomatization (+++) Automatic Regression Testing (+++) Stress Test (+++) Explorative Testing (++)
Development processes and methods:	Agile development with Scrum and Kanban (++) Waterfall (++) XP (+) Code Review [Faban, Code Collaborator, ...] (+++)
Operating systems:	Linux (+++) Solaris (+++) HP-UX (++) Unix System V (++) OS-X (Mac) (++) Windows [2000, XP, 2003, 7] (++) CP/M and Concurrent CP/M-86 (++) ORG 300 (++) AMBOSS (+) OS/2 (+)

PROJECTS

From 6/2014 (ongoing)	Development of a Web-based version of the Japanese learning-aid described below.
Details	The Python-based interactive Japanese learning-aid, which I had started end of 2013 as a Python/tkinter version, is now re-implemented as a web application. It can be found on Github on https://github.com/rovf/rkanren and is hosted at https://rkanren.herokuapp.com/ .
Technology	Ruby On Rails, SCSS
Tasks	Analysis, Design, Implementation, Test

2/2014	Developing a web page for helping calculating expenses
Details	People organizing a party at home, often want to distribute the expenses evenly over all participants. This application calculates, who has to pay how much to whom afterwards. The application can be used here: http://www.fusshuhn.de/sw/fc/SplittingUpTheBill.html
Technology	JavaScript
Tasks	Analysis, Design, Implementation, Test

1/2014	Developing a prototype for a HTML5 canvas application
Details	As part of a larger web application, which is still in the design stage, an interactive (client-side) tool needed to be implemented, involving the HTML5 Canvas element. A demo version can be seen here: http://www.fusshuhn.de/sw/canvas_demo/cantest_demo1.html
Technology	JavaScript, with some jQuery. Development is currently done on Mac OS X using the Safari, Caminion and OmniWeb web browsers.
Tasks	Analysis, Design, Implementation, Test

From 9/2013 (ongoing work)	Development of a learning-aid for learning Japanese
Details	The learning-aid helps to train japanese expressions. I'm developing this on my own, because I didn't find anything with exactly the features I needed on the market. The following pecularites of the Japanese language will be covered: × A 1:1-translation between German and Japanese words can be done only in rare cases. In general, words and expressions need to be learned according to the context (for instance, using example sentences). Memorizing words using a dictionary just doesn't make sense when learning Japanese. × A person learning the Japanese, develops over the time his individual set of example sentences. Therefore, the application must not rely on a ready-made database, but must allow the user to store his own example sentences. × While learning an Indo-Germanic language requires only two levels to be related to each other (for example, the meaning of a term in

	English, and the way to write it in German), Japanese has three of such levels, which a learner experiences as largely independent. These levels are: Meaning of the sentence in German, Reading of the sentence in Japanese (i.e. how to say it in Japanese), and the way to write it in Japanese. These levels must be supported by the application. See also: http://www.fusshuhn.de/sw/kanren/kanren.html
Technology	Python 2, Python 3. The GUI is developed using tkinter. Development is done on Mac OS X, but the software is designed to be platform independent. The program is designed to be compatible to Python 2 and 3.
Tasks	Analysis, Design, Implementation, Test

8/2013	A State-Machine as design aid.
Details	In the course of a project for a mobile application für Android, the desired system behaviour needed to be modelled somehow during the analysis phase already. In order to quickly simulate different versions of the (fairly complex) system behaviour, I implemented a configurable State Machine, which allows to quickly play various use cases, record the inputs and plays them back later (for instance, with a slightly changed state machine).
Technology	Ruby. Developed on Mac OS X, but the software is platform independent. I also got some insight into Android application development during this work.
Tasks	Analysis, Design, Implementation, Test

4/2008 until 7/2013	Design and Implementation of a Regression Test Server einer Applikation for quality assurance.
Details	In a small team of 2-4 developers (varying over the time), we developed for Infineon (the project was taken over after some time by Intel) a system fulfilling the following requirements: × Regressiontests for individual products × Integrationtests für a set of products × Calculating test metrics for automatic evaluation of the results, and automatic adjustments of the results for subsequent tests × Integration in Clearcase × Option for continuous integration × Scheduling, prioritizing and Parallelization of tests × Automatic elimination of redundant tests to reduce running time of the test × Generating HTML reports of the test results × Maintaining the test results in a database The last point was particularily important, because without this optimization, some tests would have run for a full day. My tasks in this project were: × Design, implementation and test of several subsystems of the Test Server. × Design and implementation of a component test framwork for

	testing subsystems of the Test Server. × Specification of the Integration tests for the Test Server. × Extending a stress test tool for our Test Server. × Working closely with the end-users of our Test Server, to discuss requirements for changes and new features, and to explain to them how to use the Test Server. × 3rd-Level-Support (Bug tracking, failure analysis). × Helping the end-users with their own applications, which are going to be tested using the Test Server. × Implementation of interfaces to other systems, which are used within Infineon/Intel and want to interact with our Test Server. × Development of a complex system of tools to automatically erase unneeded artifacts from previous tests, which have been left over in the database and in the file system. × Development of a tool set, which is used in emergency situations (if storage gets low) to quickly erase artifacts according to various parameters. × Development of a tool set (for example, scripts to automatically log into remote systems and perform certain tasks) to help support work. × Development of tools for automatic deployment. * Design and development of various optimization strategies, for example of a method, which represents tests as a ordered, coloured graph, and reducing this graph successively, until the minimum of tests needed to execute, is left over. * Maintaining XML Schema definitions (since the test input, and the configuration of the Test Server, was written in XML). The software for the Test Server was versioned to a large part using Clearcase. For some part of the software, Subversion and git was used instead. Perl was used as the main implementation language, and the connection to the Oracle-Database was done using DBI. A plethora of other programming- and scripting languages was also used for implementint various utilities. One interesting aspect of this projects was the multi-cultural environment, as part of the work was done in Portugal (I could make use of my knowledge in Portuguese here), others in India, Singapore and Israel.
Technology	LSF (IBM Load Sharing Facility), Perl, bash, zsh, Oracle, XML, XML Schema, HTML, Clearcase, Windows, Solaris, Linux, UTP, jEdit, Notepad++ (for validating XML documents), SQL Developer and, to a smaller extent: Tcl/Expect, Ruby, Python, CSS, svn (Subversion), ssh, PuTTY
Tasks	Analysis, Design, Implementation, Test

4/2007 until 10/2007	Design and Implementation of a testsuite for automatic regression testing.
Details	For a subproject of a NFC-application, written in Java, I developed autoated Blackbox-Regressiontests. Part of these tests were coded explicitly, but for most of them I developed a tool to generate a test program out of a template.

	I implemented these generators in Ruby (and later ported it to JRuby). They generate XML test requests, which then were processed by a test driver. I wrote the test driver in Java, and used Ant for glueing the components together. For getting at the test results, I wrote an evaluation program to query a MySql database. The Ruby version used the mysql module provided by MySql (now: Oracle). For the JRuby version, I used JDBC to interface to the database.
Technology	Ruby, JRuby, Java, JDBC, Ant, MySql, zsh, XML, XML Schema, Subversion, Windows, Linux
Tasks	Design, Implementation, Test

10/2005 until 2/2007	Design and implementation of a testsuite for automatic regression tests for embedded applications.
Details	Starting point for this project for Nokia-Siemens were a set of already existing tests scripts, which however could not be used for automatic regression testing. Not only needed these scripts manual interaction, but the results also needed to be evaluated manually. My task was to replace this manual testing system with a fully automated one. Since this was about testing embedded applications, the automated framework also needed to shuffle files to and from the remote systems and start various tasks there. For this aspect of automation, I used Perl together with Net::Telnet and various FTP libraries and tools (such as Net::FTP and sftp), as well as Tcl/Expect. From this, I developed new tests, to increase test coverage. Many of these tests were developed from technical design documents provided by the users. The regression tests could be started either from the command line, or via a control center, which I developed as a GUI application using Ruby/Tk. The application to be tested with this framework, was developed partially in München, partially in Portugal. An important aspect of the project consisted in communicating with the colleagues in Portugal. I found it helpful to already know Portuguese mentality and Portuguese language. Application and testing framework were versioned using Clearcase.
Technology	Perl, Ruby/Tk, Tcl/Expect, bash, zsh, Clearcase, Linux
Tasks	Analysis, Design, Implementation, Test

3/2003 until 6/2005	(a) Maintenance of a web presence in parallel to: (b) Automatization of deployment of a web application
Details	Both parts of the project were related to the Inter- and Intranet site of Infineon. (a) The software behind the Inter- and Intranet presence consisted at this time of a complex collection of different individual applications, written in Java and (client- and server-side-)JavaScript, using the Broadvision content management server, JSP, Struts,Ant, AspectJ, Log4J and other tools, and Apache and Microsoft IIS as web server. My task was to find and correct bugs, to react to bug reports submitted by users, and to implement new features using Java and JavaScript. (b) Deploying a new version of the site was previously done manually. This manual process was error-prone and often lead to temporary unreachability of the site. I implemented an automatic deployment tool, which ensured uninterrupted deployment, with various plausibility tests, and the automatic fall-back to the old version in case of an unrecoverale error. This was implemented in Perl and bash, with some use of Ant. Clearcase was used to for version management in both sub-projects.
Technology	Perl, bash, Java, AspectJ, JavaScript, JSP, Struts, Ant, Oracle, Clearcase, HP-UX
Tasks	Implementation, Test

3/2002 until 9/2002	Leading a project for developing a testing framework
Details	A new application to be developed in C++ at Nokia-Siemens was in need of a framework for automatic subsystem- and integrationtest. This framework was to be developed in parallel with the design of the C++ application, in close cooperation with the people responsible for the subsystem. My task as the project leader for the testing framework was to coordinate the meetings with the subprojects, and from this develop a specification for the testing framework, and its development (in particular, in reviewing all of the code - I didn't do any development myself here). The testing framework was implemented in Perl.
Technology	Perl
Tasks	Analysis, Design, Project Management

5/1999 until 12/2001	Participating in the development of a middleware-application for UMTS
Details	The communication between components of a UMTS system developed by Siemens, the Q3 protocol was generally used. In this case however, one of the components was made by a Japanese company, NEC, which used a propriatary protocol, and for mediating between these protocols, a protocol converter needed to be implemented. I was part of the team which developed this converter. The software was a multithreaded, embedded application, and was developed using C++ and the pthread library. Visual GDMOpro was used to model the Q3 requests. ASN.1 was used to define the data structures. In addition, I also had the following tasks in this project: × Developing various debugging aids (using Tcl/Expect) × Developing automatic regression tests (using Tcl/Tk and Perl) × Writing extensions for Emacs and Xemacs, which were used as the development environment in this project (ELisp) × Teaching C++ to those fellow developers who had previous experience only in C (by working out, and holding, lectures, and supervising programming exercises) × Creating programming guidelines for C++ × Working closely with the Japanese team for clarifying open issues related to the interface of their product
Technology	C++, Perl, Tcl/Tk, elisp, sh, pthreads, Q3, Visual GDMOpro, ASN.1, IBM Rational, Solaris, Unix System V
Tasks	Design, Implementation, Lecturing/Tutoring

7/1998 until 10/1998	Client-Server-Applikation for remote controling Clearcase
Details	Clearcase running under HP-UX was used by Schweizer Rentenanstalt. A group of end users was using Windows hosts only. They needed a very limited form of access to the data stored in Clearcase. Instead of installing Clearcase clients on the individual hosts, a command-line based client-server system was to be created to access Clearcase. My task was to design and implement this system. I implemented the client- and the server-side in Perl. The communication between client and server was done using sockets. Some UML was used for the design.
Technology	Perl, Clearcase, UML, HP-UX, Windows
Tasks	Analysis, Design, Implementation, Test, Project Management

3/1998 until 6/1998	(a) Refactoring of an existing information system in parallel with: (b) Developing tools for test automatization
Details	Both tasks were related to an information system made for Deutsche Telekom, which was implemented in C++ (a) The existing object hierarchy was not flexible enough to allow easy extension, and I have refactored part of it. (b) No automated tests existed yet, I implemented a simple testing framework in Perl. The C++ test driver programs were created out of templates using the M4 macro language.
Technology	C++, Perl, M4, bash, Korn Shell, SQL, Clearcase, HP-UX
Tasks	Design, Implementation, Test

10/1995 until 12/1997	(a) Development of components for an embedded application in parallel with: (b) Creating scripts for supporting testing
Details	Both tasks were related to a large telecommunication project by Siemens, for the SDH architecture. (a) I had to implement several components in C++ (analysis and design was already done by a different group). (b) Creating tools for helping to test this application. There was already in use an existing framework for testing, implemented in C++, which I extended considerably. This was not design for automatic regression testing yet, so I created tools for this too. These tools were implemented in Perl 4, Tcl and tcsh. In addition, I did some Elisp programming to extend the Emacs development environment.
Technologie	C++, Perl 4, Tcl, Emacs-LISP, tcsh, Clearcase, Solaris
Aufgabe/Funktion	Implementation, Test

SHORT OVERVIEW OF MY OLDER PROJECTS

1977 until 1995	
Details	I present here only a sketch of my activities before 1995: × Creating a C++ application running on OS/2 for doing password verification to a host running MVS. × Helping in the database design for a dance school. × Doing analysis and design for a distributed object-oriented system. CORBA was not available for this platform at this time, and I stayed as close as possible to the CORBA standard, to make an easy transition feasible as soon as we could switch to a full-blown CORBA system. The implementation was planned to be done in C++, but the project was canceled after the design phase. × Developing an embedded application in C++ for supporting network management, as part of the Siemens-Nixdorf TRANSVIEW system. × Working out (and holding) seminars for C, C++ and Oracle. × Developing a high-level widget library used for creating GUI application using Collage, which was a C++ library marketed by Siemens. × Writing an expertise about the quality of a FORTRAN 77 application, with respect to maintainablility. × Developing a converter library between CAD data formats for Siemens. The converter needed to run under Unix V and Domain OS and was implemented in Pascal and C. × Design and implementation of a front-end to the extensible "Witch System" word-processing system, which made it accessible for visually-impaired users. The implementation was done in the proprietary languages HK und WitchDOS). × Giving advice to a software company in Zurich (Switzerland) for the design of user interfaces. × Creating a MSDOS application for an insurance broker, to help comparing insurance companies. I did analysis and design, and implemented the application in the programming language NIAL. × Extending a systems for configuring airplanes, for Deutsche Airbus. The language used was APL, running under MVS. × Developing an information system for supporting firefighters at large buildings or industrial plants. Ths system was marketed by Siemens. I did a part of the design, and had project management responsibility for a subsystem of the application. It was implemented in C, Pascal and Assembler, for Concurrent CP/M-86, and was later ported to Concurrent DOS. × Extending a system for running the central control office of police-

	and fire departments. The system was marketed by Siemens, and was used in several cities in Europe and the Near East. ORG 300 and AMBOSS were used as operating systems. I wrote the larger part of the application in in Assembler, using the MAS 300 macro language as a high-level abstraction, and also developed new macros for better support of structured programming in Assembler. A subsystem of the application - a special editor for drawing floor plans of buildings - was written in Pascal. × Developing various tools for a concert agent in Austria, using a shell-like command language for MSDOS, and some SNOBOL4. × Developing a data management system for really small computers (48KB main memory, Z80 CPU). It consisted of a flat table data storage, and a query language based on wild-card patterns. The implementation was done in FORTAN IV and Assemler, running on CP/M and MP/M. × Developing an interpreter for a special-purpose version of BASIC for the ISIS operating system, to be used for controlling NC hardware. The implementation was done in 8080 Assembler. × Extending an existing FORTRAN IV application for calculating costs for architecture projects. The software was running on Data General's RDOS operating system. × Publication of various IT related articles in various magazines, for instance Mini Micro Magazin, Dr. Dobb's Journal, or Personal Computer.