

Henrik Kniberg - Portfolio



[Downloadable/printable version](#)



This page summarizes some of the projects and systems that I've had key roles in. The list is ordered by approximate complexity (not chronologically).

- [Scrum and XP from the Trenches](#)
- [Gaming system](#)
- [E-commerce engine](#)
- [XML language for mobile services](#)
- [Phone to PC remote access](#)
- [Flight booking systems](#)
- [Sms & messaging gateways](#)
- [Skinning engines](#)
- [Watchdogs](#)
- [Billing systems](#)
- [Single-signon system](#)
- [Java lab tool](#)
- [Disk cleaner tool](#)
- [Studio booking system](#)
- [Band management service](#)

CONTACT INFO

Email: henrik.kniberg@crisp.se

Skype: henrikkniberg

Phone: +46 70 4925284

LINKS

CV:

henrik.kniberg.com/cv.html

Personal Homepage:

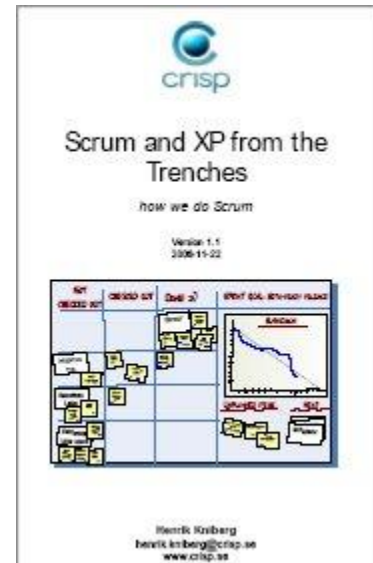
www.kniberg.com/henrik

SCRUM AND XP FROM THE TRENCHES

During my period as chief of development at [Tain](#) I wrote the book [Scrum and XP from the Trenches - How we do Scrum](#). It summarizes lessons learned from implementing Scrum and XP in a 40 person development team over a 1.5 year period.

To my pleasant surprise, the book was downloaded approximately 30,000 times within 3 months and has been mentioned in dozens of blogs and web sites as one of the best books on agile software development. Who would've guessed that? :o)

The book is currently being published at [InfoQ](#), with forwards from [Mike Cohn](#) (author of Agile Estimating and Planning) and [Jeff Sutherland](#) (co-founder of Scrum).



GAMING SYSTEM

A distributed system for casino-style gaming terminals running games such as blackjack, roulette, and super7. The system was designed for over 50,000 terminals simultaneously connected to a central system, handling billions of euros per year in monetary transactions.

Highlights:

- **Game developer's kit** - a programmer-friendly kit is provided for game developers, with all necessary APIs and tools. Game developers are cleanly isolated from the internal platform design, including all network and database communication.
- **Robust error handling** - if a terminal is disconnected in mid-game, the player receives a voucher and can continue on any other terminal.
- **Scalability** - clustering solution for servers as well as databases.
- **Administration** - centrally administration via a web interface, on-site administration via dedicated storekeeper terminals at each site.
- **Plugin architecture** - new games can be added to the system on the fly.
- **Central configuration** - all terminals can be administered centrally, no need to travel to the gaming venues.
- **High security** - all traffic is encrypted, all security-sensitive operations are performed on the server-side.
- **Version handling** - all key APIs and protocols are version-handled for future-compatibility as well as backward-compatibility. Advanced schemes for version conflicts between games, protocols, etc.



Technology:

This is large system with the different subsystems developed using different technologies. The following technologies were used (among others): Java, JUnit, Ant, C#, C++, XML, P/SQL, Oracle, and Websphere.

E-COMMERCE ENGINE

YadaFuel is an e-commerce engine that today handles millions of € per year in B2B and B2C transactions, mainly in prepaid vouchers for mobile phones. This system maintains connections to multiple content providers and sells electronic vouchers through a large network of partners and distribution channels.

Multiple distribution channels:

- **Web store.** This is a skinnable, cobrandable solution where existing web sites can integrate YadaFuel as part of their overall solution. This is currently available on a large number of well-established Swedish web sites such as MSN and Aftonbladet.
- **Card terminals.** Retail chains sell electronic vouchers through their credit card terminals. The terminal connects to YadaFuel securely over the phone network, downloads vouchers, and prints them on the terminal receipt.
- **Retail store PC.** Retail chains can use a web interface to fetch vouchers from YadaFuel and print on the local printer, or topup the user's phone directly.
- **Mobile phone.** Users can make a call (from their own phone or a friend's) and order a prepaid voucher to be sent via SMS.

Multiple payment systems:

- **Cash** (for over-the-counter purchases in retail stores)
- **Credit card**
- **Internet bank** (i.e. user is redirected to internet bank to make a payment, and then redirected back to the web store)
- **Autogiro** (i.e. user sets up an Autogiro payment relationship allowing us to deduct his bank account after each purchase)
- **Invoice** (for clearing with retail chains)

Other features:

- **Partner web interface** - Partners can log in to a web interface to track purchase statistics.
- **Advanced helpdesk system** where helpdesk personell could find out all necessary information about a

- purchase and perform operations such as resending a vouchers or blocking suspicious users.
- **Security system** based on detection of suspicious activity and automatic blocking mechanisms, and integration with third-party security providers.
- **Plugin architecture** for content provider connection protocols.
- **Skinning engine**, allowing user interfaces to be adapted for each partner.

Technology: YadaFuel is built as a set of interoperable EJB components with an underlying RDB. A set of Java web applications (JSP/Servlets/Struts) are used to render the web interfaces, XSL is used for skinning. Security is handled through a variety of protocols including public key cryptography, symmetric encryption of data communication, and digital signatures for tamper-protection. JMS is used for transaction logging and sms communication. Also used JUnit and Ant.

SMURF - AN XML LANGUAGE FOR MOBILE SERVICES

"Smurf" is an XML based language for rapid development of interactive mobile services. Smurf stands for SMS-SURF. I designed the language, the engine that interprets it, and development tools for creating and testing smurf services.

Smurf allows you to create mobile services in much the same way that you create web sites, i.e. by a set of pages that refer to each other through links. So instead of surfing around on html page using a web browser, the user "surfs" around on smurf pages using his mobile phone. Each smurf page corresponds to an sms, and the user "clicks" on a link by sending an sms or calling a phone number and hanging up after 1 signal.

Through a plugin architecture it is possible to add "actions", for example a "billing" action can be used to deduct a user account, a "send email" action can be used to send an email, and a "http" action can be used to trigger a request to a remote system over http.

Smurf was patented in 1999 and has been put to use in a number of applications, for example:

- **Mobile quizzes**, where users answer a set of questions in order to win prizes.
- **Voting services**, for example in one case users would vote for their favorite hockey player during matches.
- **Multiplayer games**, for example adventure style games where the user moves through a set of rooms and can interact with other players in the same room.
- **Sales statistics**, i.e. management could access sales information through their mobile phones.
- **Server monitoring**, i.e. access to server status information and simple administrative functions through the mobile phone.

Technology: EJB, XML, JMS, Servlets, JSP, OODB, RDB, Struts, JUnit, Ant, SMS.

PHONE TO PC REMOTE ACCESS

This is a patented architecture that enables your mobile phone to be used as a remote access device for your PC, regardless of your location. This allows remote access to files stored on your PC or remote control of your PC using your mobile phone.

Features:

- **Easy installation** - the user only has to download a program and enter his phone number.
- **Firewallproof** - works even if the PC is behind a firewall and has a private IP address.
- **Automatic content conversion** - Content is adapted automatically to the specific constraints of your phone model.
- **Platform-independent** - works on PC, Mac, Linux, etc.
- **Flexible licensing model** allowing distribution through partners and adapting to their licensing models.
- **Scalable** - the server is clusterable.
- **Skinnable client** - the appearance of the client in the PC can be adapted for different partners.

Technology: All components are Javabased. The server is in fact an http server, written from scratch and customized for this architecture. The client is a swing



application which includes a built-in MP3 players. Network communication and configuration management is done using XML. Mobile phone clients are written as Java midlet, wml, or xhtml depending on the application and mobile phone model. Image manipulation is done via JAI and Java2D. Also used JUnit and Ant.



FLIGHT BOOKING SYSTEMS

A suite of flight booking systems used by [Skyways](#). My biggest project was a flight booking web application for travel agents including web design, integration with payment providers, database design, requirements analysis, etc. At this company I introduced modern development techniques such as [continuous integration](#) and [test-driven development](#), and modern frameworks such as [Hibernate](#) and [Spring framework](#).

Technology: Java, Spring, Hibernate, Web services, XML, JSP, Servlets, JUnit, HttpUnit, jWebUnit, Ant, AntHill, PostgreSQL,



SMS & MESSAGING GATEWAYS

Since both Goyada and Epicent use SMS for a variety of purposes I've managed the development of several generations of SMS gateways for both companies.

Key features:

- **Multiple connection protocols.** SMS-handling applications can connect through a variety of protocols including http, jms, and rmi.
- **Plugin architecture** allowing connections to different types of sms providers using different protocols.
- **Failover rules** - if one sms provider fails another will be used automatically.
- **Traffic priority rules**, ensuring load balance between sms connections.
- **Web based administration** interface allowing you to stop/start connections, install plugins, search messages, create traffic reports, and configure routing rules.
- **24/7 operation** - will keep running even under the most unstable conditions, and ask administrators for assistance when necessary.
- **Fault detection rules** - for example could trigger alarm if a certain sms account does not receive at least 120 messages per hour.

Technology: Java, JMS, Sybase, RMI, HTTP, JSP/Servlets, Struts, various smsc protocols, Junit, Ant.

SKINNING ENGINES

Both Goyada and Epicent make significant use of skinning, or cobranding. This means taking one application and giving it different "faces" (or "skins") to blend in with different partners' web sites ("cobranding"). With a significant number of partners, each with their own particular demands on layout, this quickly becomes a maintenance headache. Unless of course you have a good skinning engine.

I've designed skinning engines on several occasions, including features such as:

- **Seperation of content and presentation.** All texts and other dynamic content are stored in a database.
- **Language flexibility.** Choice of language is done dynamically based on the web site or the user's web browser configuration.
- **Hierarchical skins.** Each skin can have a "parent" skin from which it inherits all layout, style sheets, texts,

pictures, languages, etc. This site inheritance tree allows new skins to be created with an absolut minimum amount of work. Supports requests like "I want site Y to look exactly like site X, but with red buttons and a different disclaimer text".

- **XSL transformations**, allowing the basic service structure to be described using XML, and then transformed to the appropriate layout using XSL.

Technology: EJB, RDB, Servlets, JSP, Taglibs, XSL, JUnit, Ant.

WATCHDOGS

I'm a firm believer that servers should manage themselves. Almost anything that an administrator does to check the status of a server can be automated. Only in rare cases should an administrator have to be involved, and in those cases he should be notified automatically. This not only keeps your tech team and managers sleeping better at night, it also increases your quality of service by making sure that the right person is notified (on his phone of course) the instant something serious goes wrong.

Ever since 1995 I've continually developed and polished the concept of automated monitoring and alarm handling. This has resulted in watchdog applications with the following features:

- **Configurable rules** for:
 - **Which servers** should be checked and how
 - **Which people** should be notified in which cases
 - **Different tolerans rates** at different times of day ("if server X runs into a problem at 3 am and fixes it by itself within 5 minutes then don't wake up the administrator! ")
 - **Which communication medium** should be used in which cases, for example sms or email.
- **Plugin architecture** for test execution protocols. Includes:
 - **http** checks - the watchdog issues a web request and checks for an expected response.
 - **junit** checks, the watchdog runs a junit test suite.
- **Web admin interface** that is accessible from PCs or mobile phones.
- **Swing-based graphical user interface.**
- **Distributed architecture** allowing watchdogs to monitor each other.
- **Reporting** facilities.

Technology: Swing, Servlets, JSP, XML, SMS, XHTML, JUnit, Ant.

BILLING SYSTEMS

Over the years I've designed several billing systems that support multiple business models. This includes features such as:

- **Web administration interface**, where you can configure products, prices, agreement rules, invoice generation rules, and statistics logging rules.
- **Partner login**, i.e. revenue share partners can log in themselves through a web interface and see transaction data and revenue summaries based on partner-specific agreement rules.
- **Plugin architecture** for agreement rules. As your business model evolves you can encode them as plugins to the billing system in order to generate correctly calculated invoices and reports.
- **Asynchronous** logging, so that service performance is not hampered by having to wait each time a transaction is logged to the database.

Technology: EJB, RDB, Servlets, JSP, Struts, JUnit, Ant.

SINGLE-SIGNON SYSTEM FOR JAVA APPLETS

This is a set of JavaBeans components that make it easy to provide single-signon functionality for applets. I.e. a company with many different types of applets can use this system to remove the need for employees to log in seperately for each applet.

This included:

- Digitally signed components enabling the login components to escape the browser's security sandbox in order to communicate with the login server.
- Client-side components that display login windows, capture and validate input, communicate with the server and fire JavaBeans events to notify the rest of the applet when a user has logged in or out

- Server-side components that handle communication with the client and authentication of users against backend systems.
- A servlet proxy solution that makes it possible for client components to communicate with server components through a servlet, in cases where signed beans don't work.

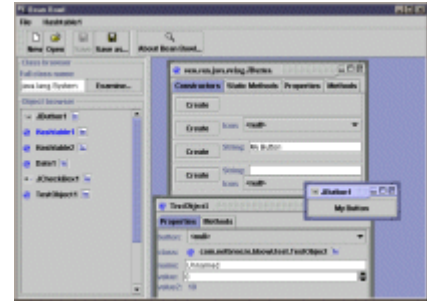
Technology: Applets, JavaBeans, Swing, AWT, Servlets, JNI.

BEANBOWL - A JAVA LAB TOOL

Bean Bowl is an application that allows you to work with Java classes and objects in "real time", using direct manipulation techniques. It lets you study the behaviour of programs, classes, and APIs without having to write any code at all.

I developed Bean Bowl 1995 but due to popular interest I released Bean Bowl as open source project after a couple of years. It is available at <http://beanbowl.sourceforge.net>.

Technology: Swing, JavaBeans, Java reflection, Ant. APIs.

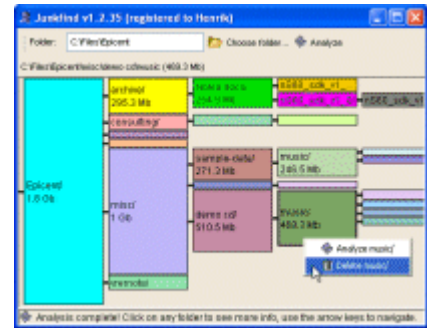


JUNKFIND - A DISK CLEANER TOOL

Junkfind is a high-level disk analysis and cleanup tool. It analyzes the content of any disk or folder and shows a graphical map of the contents. The map makes it easy to see what is taking most space without having to dig around inside folders, and lets you remove folders through a right-click menu.

Junkfind is available at <http://junkfind.kniberg.com>.

Technology: Java Webstart, Swing, JUnit, Ant, Java2D.



STUDIO BOOKING SYSTEM

This is a web-based booking system for a rehearsal and recording studio in Stockholm. Through heavy use of javascript the user interface is highly responsive and easy to use, which is important since most users are musicians with little technical knowledge.

Technology: Javascript, Spring, Tapestry, XML, JSP, JUnit, Ant.



BANDPLANNER - A BAND MANAGEMENT SERVICE

Bandplanner is a web-based meeting planner. It is designed to make it easy for bands, orchestras, theatre groups, and other similar communities to find good rehearsal days.

Bandplanner is available at <http://bandplanner.kniberg.com>.

Technology: EJB, RDB, XML, JSP, Servlets, Struts, JUnit, Ant.

