

Agile Secure Software Development in a Large Software Development Organisation

Security Testing

Achim D. Brucker

achim.brucker@sap.com

<http://www.brucker.ch/>

SAP SE, Vincenz-Priessnitz-Str. 1, 76131 Karlsruhe, Germany

ASSD Keynote

First International Workshop on Agile Secure Software Development (ASSD)

Toulouse, France, August 24–28, 2015

Agile Secure Software Development in a Large Software Development Organisation

Security Testing

Abstract

Security testing is an important part of any (agile) secure software development lifecycle. Still, security testing is often understood as an activity done by security testers in the time between “end of development” and “offering the product to customers.”

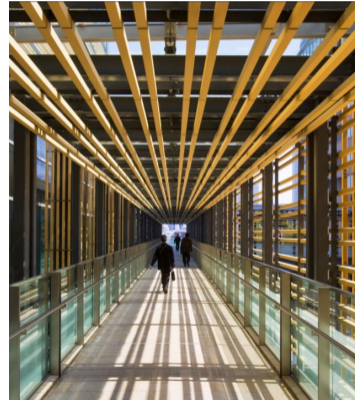
Learning from traditional testing that the fixing of bugs is the more costly the later it is done in development, we believe that security testing should be integrated into the daily development activities. To achieve this, we developed a security testing strategy, as part of SAP’s security development lifecycle which supports the specific needs of the various software development models at SAP.

In this presentation, we will briefly presents SAP’s approach to an agile secure software development process in general and, in particular, present SAP’s Security Testing Strategy that enables developers to find security vulnerabilities early by applying a variety of different security testing methods and tools.

Agenda

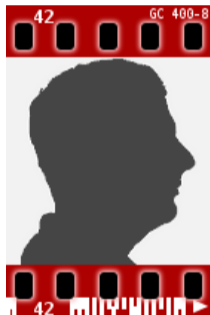
- 1 Background
- 2 Motivation
- 3 Risk-based Security Testing as Part of SAP's S²DL
- 4 Lesson's Learned
- 5 How Does This Resonate With Agile Development?

- Leader in Business Software
 - Cloud
 - Mobile
 - On premise
- Many different technologies and platforms, e.g.,
 - In-memory database and application server (HANA)
 - Netweaver for ABAP and Java
- More than 25 industries
- 63% of the world's transaction revenue touches an SAP system
- over 68 000 employees worldwide
over 25 000 software developers
- Headquarters: Walldorf, Germany (close to Heidelberg)



Personal Background

- I wear two hats:
 - **(Global) Security Testing Strategist**
 - Research Expert/ArchitectWorking for the central software security team
- Background:
Security, Formal Methods, Software Engineering
- Current work areas:
 - Static code analysis
 - (Dynamic) Security Testing
 - Mobile Security
 - Security Development Lifecycle
 - Secure Software Development Lifecycle



<http://www.brucker.ch/>

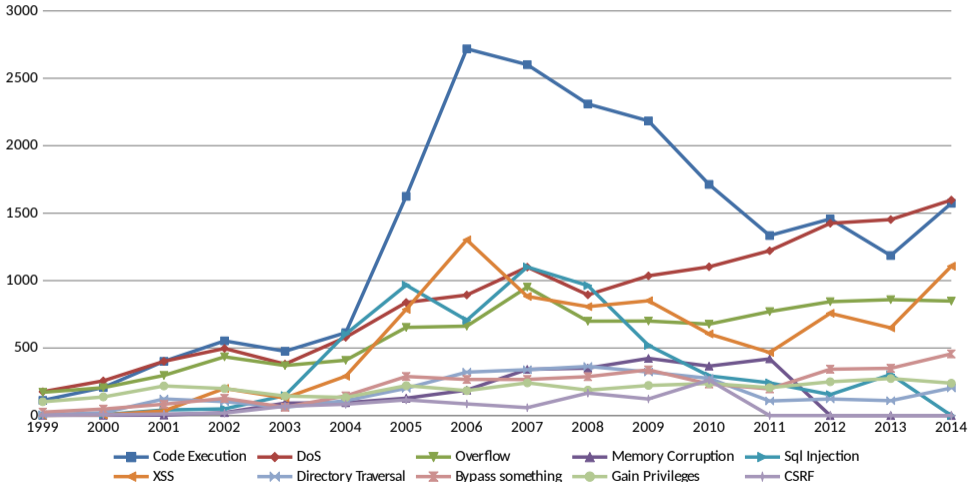
SAP Uses a De-centralised Secure Development Approach

- **Central security expert team** (S²DL owner)
 - Organizes security trainings
 - Defines product standard “Security”
 - Defines risk and threat assessment methods
 - Defines security testing strategy
 - Selects and provides security testing tools
 - Validates products
 - Defines and executes response process
- **Local security experts**
 - Embedded into development teams
 - Organize local security activities
 - Support developers and architects
 - Support product owners (responsibles)
- **Development teams**
 - Select technologies
 - Select development model
 - Design and execute security testing plan
 - ...

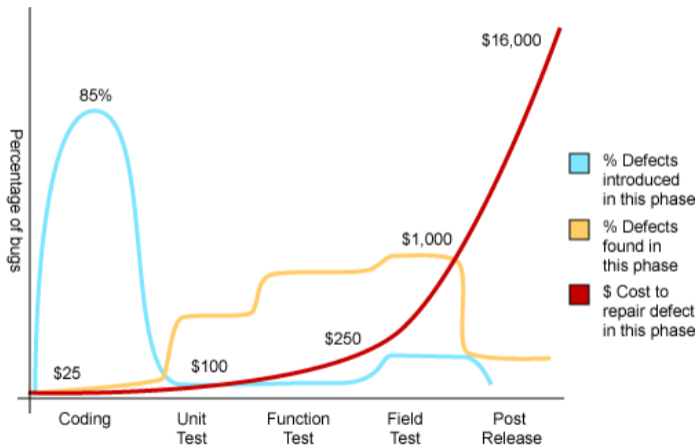
Agenda

- 1 Background
- 2 Motivation**
- 3 Risk-based Security Testing as Part of SAP's S²DL
- 4 Lesson's Learned
- 5 How Does This Resonate With Agile Development?

Vulnerability Distribution

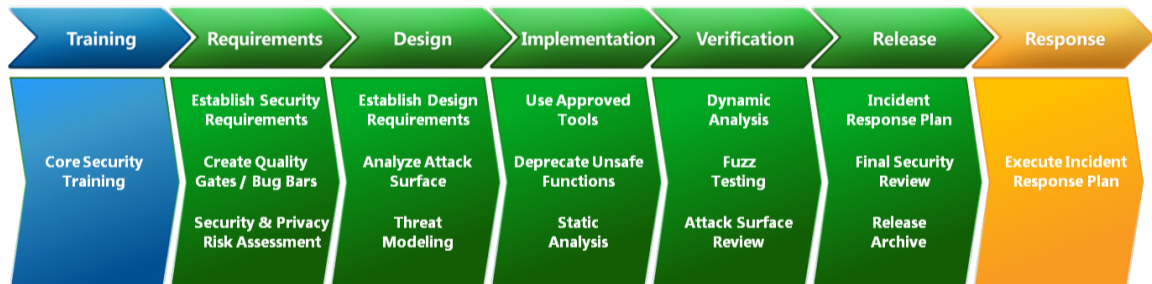


When Do We Fix Bugs



Source: Applied Software Measurement, Capers Jones, 1996

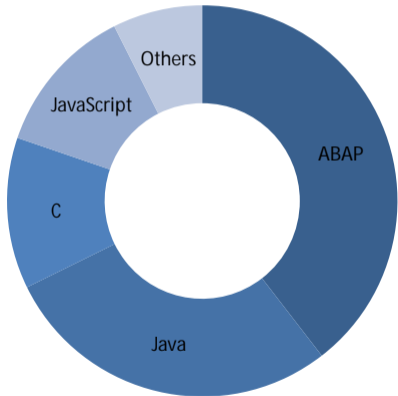
Microsoft's SDL



Agenda

- 1 Background
- 2 Motivation
- 3 Risk-based Security Testing as Part of SAP's S²DL**
- 4 Lesson's Learned
- 5 How Does This Resonate With Agile Development?

Our Start: SAST as a Baseline



SAST tools used at SAP:

Language	Tool	Vendor
ABAP	CVA (SLIN_SEC)	SAP
JavaScript	Checkmarx CxSAST	Checkmarx
C/C++	Coverity	Coverity
Others	Fortify	HP

- Since 2010, mandatory for all SAP products
- Multiple billions lines analyzed
- Constant improvement of tool configuration
- Further details:
Deploying Static Application Security Testing on a Large Scale. In GI Sicherheit 2014. Lecture Notes in Informatics, 228, pages 91-101, GI, 2014.

Combining Multiple Security Testing Methods and Tools



Client Application

Web Browser

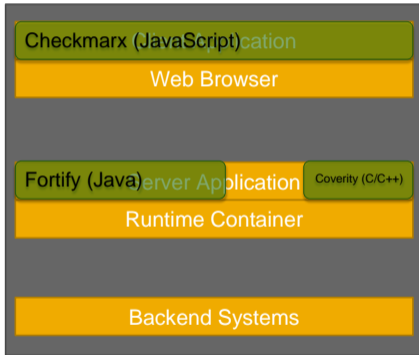
Server Application

Runtime Container

Backend Systems

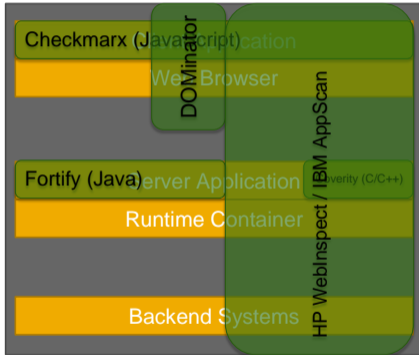
- Risks of only using only SAST
 - Wasting effort that could be used more wisely elsewhere
 - Shipping insecure software
- Examples of SAST limitations
 - Not all programming languages supported
 - Covers not all layers of the software stack

Combining Multiple Security Testing Methods and Tools



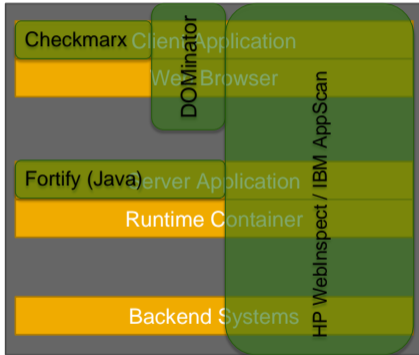
- Risks of only using only SAST
 - Wasting effort that could be used more wisely elsewhere
 - Shipping insecure software
- Examples of SAST limitations
 - Not all programming languages supported
 - Covers not all layers of the software stack

Combining Multiple Security Testing Methods and Tools



- Risks of only using only SAST
 - Wasting effort that could be used more wisely elsewhere
 - Shipping insecure software
- Examples of SAST limitations
 - Not all programming languages supported
 - Covers not all layers of the software stack

Combining Multiple Security Testing Methods and Tools



- Risks of only using only SAST
 - Wasting effort that could be used more wisely elsewhere
 - Shipping insecure software
- Examples of SAST limitations
 - Not all programming languages supported
 - Covers not all layers of the software stack

A Risk-based Test Plan



- Combines multiple security testing methods, e.g., code scans, dynamic analysis, manual penetration testing or fuzzing
- Selects the most efficient test tools and test cases based on the risks and the technologies used in the project
- Re-adjusts priorities of test cases based on identified risks for the project
- Monitors false negative findings in the results of risk assessment

SAP' Secure Software Development Lifecycle (S²DL)

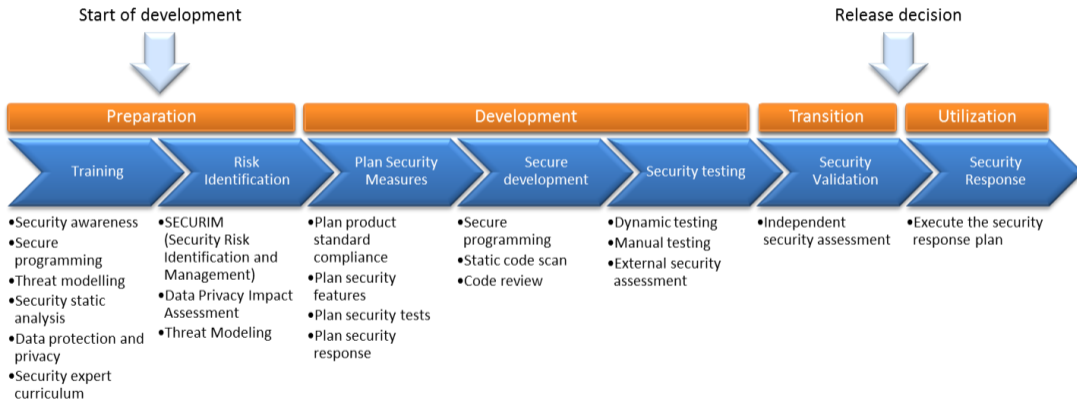


Figure: SAP SSDL

Security Validation

- Acts as first customer
- Is not a replacement for security testing during development
- Security Validation
 - Check for “flaws” in the implementation of the S²DL
 - Ideally, security validation finds:
 - No issues that can be fixed/detected earlier
 - Only issues that cannot be detect earlier
(e.g., insecure default configurations, missing security documentation)

Security Validation

- Acts as first customer
- Is not a replacement for security testing during development
- Security Validation
 - Check for “flaws” in the implementation of the S²DL
 - Ideally, security validation finds:
 - No issues that can be fixed/detected earlier
 - Only issues that cannot be detect earlier
(e.g., insecure default configurations, missing security documentation)

Penetration tests in productive environments are different:

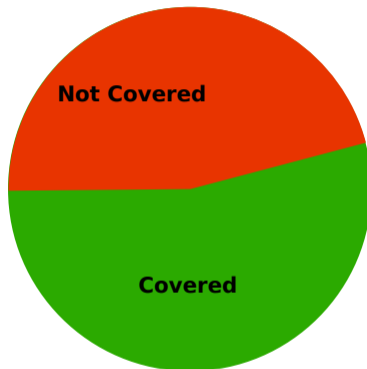
- They test the actual configuration
- They test the productive environment (e.g., cloud/hosting)

How to Measure Success

- Analyze the vulnerabilities reported by
 - Security Validation
 - External security researchers
- Vulnerability not detected by our security testing tools
 - Improve tool configuration
 - Introduce new tools
- Vulnerability detected by our security testing tools
 - Vulnerability in older software release
 - Analyze reason for missing vulnerability

How to Measure Success

- Analyze the vulnerabilities reported by
 - Security Validation
 - External security researchers
- Vulnerability not detected by our security testing tools
 - Improve tool configuration
 - Introduce new tools
- Vulnerability detected by our security testing tools
 - Vulnerability in older software release
 - Analyze reason for missing vulnerability

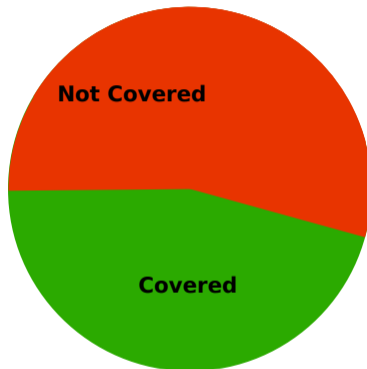


Success criteria:

Percentage of vulnerabilities not covered by our security testing tools increases

How to Measure Success

- Analyze the vulnerabilities reported by
 - Security Validation
 - External security researchers
- Vulnerability not detected by our security testing tools
 - Improve tool configuration
 - Introduce new tools
- Vulnerability detected by our security testing tools
 - Vulnerability in older software release
 - Analyze reason for missing vulnerability

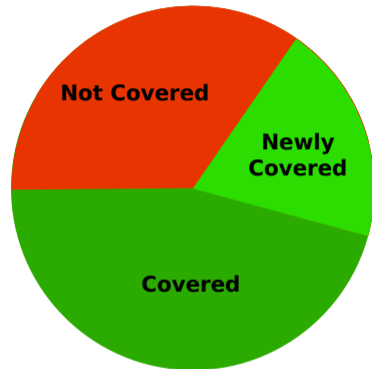


Success criteria:

Percentage of vulnerabilities not covered by our security testing tools increases

How to Measure Success

- Analyze the vulnerabilities reported by
 - Security Validation
 - External security researchers
- Vulnerability not detected by our security testing tools
 - Improve tool configuration
 - Introduce new tools
- Vulnerability detected by our security testing tools
 - Vulnerability in older software release
 - Analyze reason for missing vulnerability



Success criteria:

Percentage of vulnerabilities not covered by our security testing tools increases

Agenda

- 1 Background
- 2 Motivation
- 3 Risk-based Security Testing as Part of SAP's S²DL
- 4 Lesson's Learned**
- 5 How Does This Resonate With Agile Development?

Key Success Factors

- A holistic security awareness program for
 - Developers
 - Managers

Key Success Factors

- A holistic security awareness program for
 - Developers
 - Managers
- Yes, security awareness is important

Key Success Factors

- A holistic security awareness program for
 - Developers
 - Managers
- Yes, security awareness is important **but**

Key Success Factors

- A holistic security awareness program for
 - Developers
 - Managers
- Yes, security awareness is important **but**

Developer awareness is even more important!

Listen to Your Developers!

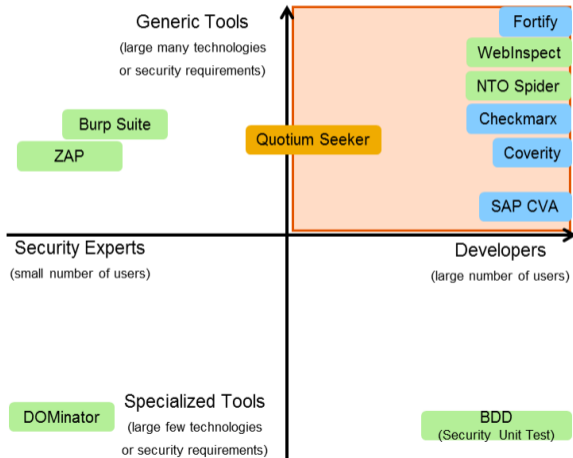
We are often talking about a lack of security awareness and, by that, forget the problem of lacking development awareness.

- Building a secure system more difficult than finding a successful attack.
- Do not expect your developers to become penetration testers (or security experts)!

Security Testing for Developers

Security testing tools for developers, need to

- Be applicable from the start of development
- Automate the security knowledge
- Be deeply integrated into the dev. env., e.g.,
 - IDE (instant feedback)
 - Continuous integration
- Provide easy to understand fix recommendations
- Declare their “sweet spots”



Collaborate!

Security experts need to collaborate with development experts to

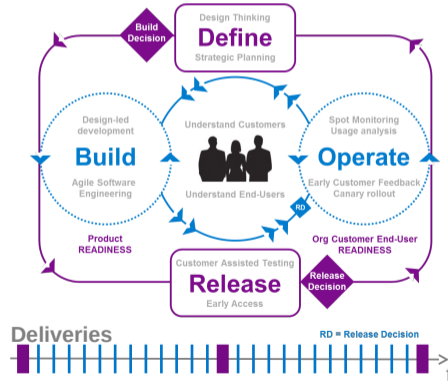
- Create easy to use security APIs (ever tried to use an SSL API securely)
- Create languages and frameworks that make it hard to implement insecure systems
- Explain how to program securely

Agenda

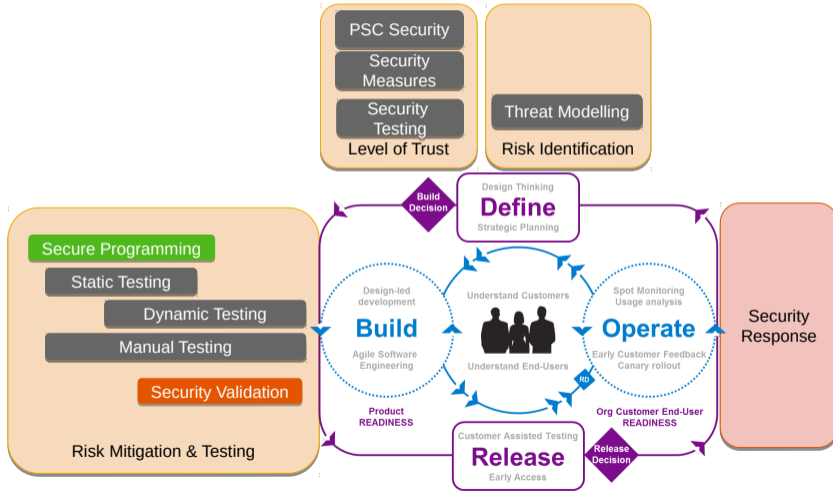
- 1 Background
- 2 Motivation
- 3 Risk-based Security Testing as Part of SAP's S²DL
- 4 Lesson's Learned
- 5 How Does This Resonate With Agile Development?**

Agile Development

- What is agile for you?
SCRUM, Continuous Delivery, DevOps, SCRUM, Cloud development, ...
- Cloud/agile development lifecycle



Secure Agile Development



Open (Research) Questions

- Social aspects
 - Does the SecDevOps model increase security awareness?
(Developers and their managers are also responsible for operational risks)
 - Does this impact the willingness to take (security) risks and/or the risk assessment?
- Process and organisational aspects
 - What services should be offered centrally?
 - How to ensure a certain level of security across all products?
 - How to ensure a certain level of security across the end-to-end supply chain?
- Technical and fundamental aspects
 - How do we need to adapt development support
 - How do we need to adapt threat modelling or risk assessment methods
 - How do we need to adapt security testing techniques
- The big challenge in practice:
Products are often offered in the cloud (SaaS) and on premise

Thank you!



<http://xkcd.com/327/>

Dr. Achim D. Brucker
achim.brucker@sap.com
<http://www.brucker.ch/>

Related Publications



Ruediger Bachmann and Achim D. Brucker.

Developing secure software: A holistic approach to security testing.

Datenschutz und Datensicherheit (DuD), 38(4):257–261, April 2014.

<http://www.brucker.ch/bibliography/abstract/bachmann.ea-security-testing-2014>.



Achim D. Brucker, Lukas Brügger, and Burkhart Wolff.

Formal firewall conformance testing: An application of test and proof techniques.

Software Testing, Verification & Reliability (STVR), 25(1):34–71, 2015.

<http://www.brucker.ch/bibliography/abstract/brucker.ea-formal-fw-testing-2014>.



Achim D. Brucker and Uwe Sodan.

Deploying static application security testing on a large scale.

In Stefan Katzenbeisser, Volkmar Lotz, and Edgar Weippl, editors, *gi Sicherheit 2014*, volume 228 of *Lecture Notes in Informatics*, pages 91–101. *gi*, March 2014. ISBN 978-3-88579-622-0.

<http://www.brucker.ch/bibliography/abstract/brucker.ea-sast-experiences-2014>.



Achim D. Brucker and Burkhart Wolff.

On theorem prover-based testing.

Formal Aspects of Computing (FAC), 25(5):683–721, 2013.

ISSN 0934-5043.

<http://www.brucker.ch/bibliography/abstract/brucker.ea-theorem-prover-2012>.

No part of this publication may be reproduced or transmitted in any form or for any purpose without the express permission of SAP SE. The information contained herein may be changed without prior notice.

Some software products marketed by SAP SE and its distributors contain proprietary software components of other software vendors.

Microsoft, Windows, Excel, Outlook, and PowerPoint are registered trademarks of Microsoft Corporation.

IBM, DB2, DB2 Universal Database, System i, System i5, System p, System p5, System x, System z, System z10, System z9, z10, z9, iSeries, pSeries, xSeries, zSeries, eServer, z/VM, z/OS, i5/OS, S/390, OS/390, OS/400, AS/400, S/390 Parallel Enterprise Server, PowerVM, Power Architecture, POWER6+, POWER6, POWER5+, POWER5, POWER, OpenPower, PowerPC, BatchPipes, BladeCenter, System Storage, GPFS, HACMP, RETAIN, DB2 Connect, RACF, Redbooks, OS/2, Parallel Sysplex, MVS/ESA, AIX, Intelligent Miner, WebSphere, Netfinity, Tivoli and Informix are trademarks or registered trademarks of IBM Corporation.

Linux is the registered trademark of Linus Torvalds in the U.S. and other countries.

Adobe, the Adobe logo, Acrobat, PostScript, and Reader are either trademarks or registered trademarks of Adobe Systems Incorporated in the United States and/or other countries.

Oracle is a registered trademark of Oracle Corporation.

UNIX, X/Open, OSF/1, and Motif are registered trademarks of the Open Group.

Citrix, ICA, Program Neighborhood, MetaFrame, WinFrame, VideoFrame, and MultiWin are trademarks or registered trademarks of Citrix Systems, Inc.

HTML, XML, XHTML and W3C are trademarks or registered trademarks of W3C®, World Wide Web Consortium, Massachusetts Institute of Technology.

Java is a registered trademark of Sun Microsystems, Inc.

JavaScript is a registered trademark of Sun Microsystems, Inc., used under license for technology invented and implemented by Netscape.

SAP, R/3, SAP NetWeaver, Duet, PartnerEdge, ByDesign, SAP BusinessObjects Explorer, StreamWork, and other SAP products and services mentioned herein as well as their respective logos are trademarks or registered trademarks of SAP SE in Germany and other countries.

Business Objects and the Business Objects logo, BusinessObjects, Crystal Reports, Crystal Decisions, Web Intelligence, Xcelsius, and other Business Objects products and services mentioned herein as well as their respective logos are trademarks or registered trademarks of Business Objects Software Ltd. Business Objects is an SAP company.

Sybase and Adaptive Server, iAnywhere, Sybase 365, SQL Anywhere, and other Sybase products and services mentioned herein as well as their respective logos are trademarks or registered trademarks of Sybase, Inc. Sybase is an SAP company.

All other product and service names mentioned are the trademarks of their respective companies. Data contained in this document serves informational purposes only. National product specifications may vary.

The information in this document is proprietary to SAP. No part of this document may be reproduced, copied, or transmitted in any form or for any purpose without the express prior written permission of SAP SE.

This document is a preliminary version and not subject to your license agreement or any other agreement with SAP. This document contains only intended strategies, developments, and functionalities of the SAP® product and is not intended to be binding upon SAP to any particular course of business, product strategy, and/or development. Please note that this document is subject to change and may be changed by SAP at any time without notice.

SAP assumes no responsibility for errors or omissions in this document. SAP does not warrant the accuracy or completeness of the information, text, graphics, links, or other items contained within this material. This document is provided without a warranty of any kind, either express or implied, including but not limited to the implied warranties of merchantability, fitness for a particular purpose, or non-infringement.

SAP shall have no liability for damages of any kind including without limitation direct, special, indirect, or consequential damages that may result from the use of these materials. This limitation shall not apply in cases of intent or gross negligence.

The statutory liability for personal injury and defective products is not affected. SAP has no control over the information that you may access through the use of hot links contained in these materials and does not endorse your use of third-party Web pages nor provide any warranty whatsoever relating to third-party Web pages.