

Towards Developing Secure Systems using UML

Jan Jürjens

Computing Laboratory, University of Oxford

jan@comlab.ox.ac.uk

<http://www.jurjens.de/jan>

Motivation

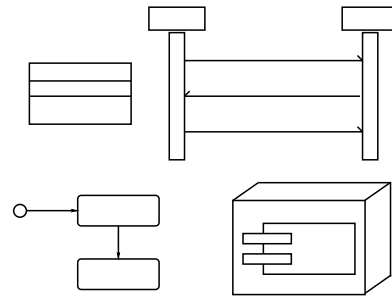
Security important (business transactions over Internet), but developers often lack background in security.

Cannot use security mechanisms “blindly”:
Security often compromised by **circumventing** them.

Encapsulate knowledge on **prudent security engineering** in UML to aid secure systems development.

Here: information flow, protocols

UMLsec (fragment)



- **Statechart diagram:** secure information flow within object
- **Class diagram:** exchange of data preserves security levels
- **Sequence diagram:** correctness of security-critical interaction
- **Deployment diagram:** physical security requirements

Statechart diagrams

Statechart diagram S interpreted as set $\llbracket S \rrbracket$ of functions from sequences of input events to sequences of output actions.

Data may be “high” or “low”.

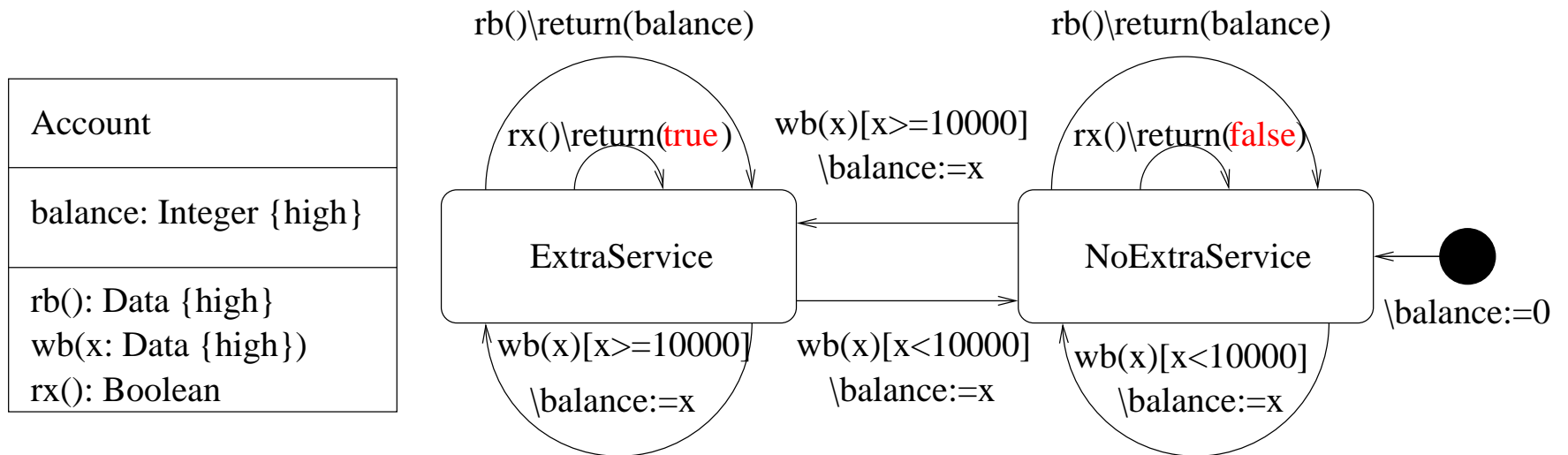
Low view of a sequence: Substitute **high** data by \square .

Object with statechart diagram S gives **preserves security** if for every $h \in \llbracket S \rrbracket$, all \vec{e}, \vec{f}

$$\mathcal{L}(\vec{e}) = \mathcal{L}(\vec{f}) \Rightarrow \mathcal{L}(h(\vec{e})) = \mathcal{L}(h(\vec{f}))$$

(Noninterference (Goguen, Meseguer 82))

Example: Entry in multi-level database



This object does not preserve security
 (`rx()` leaks information on account balance).

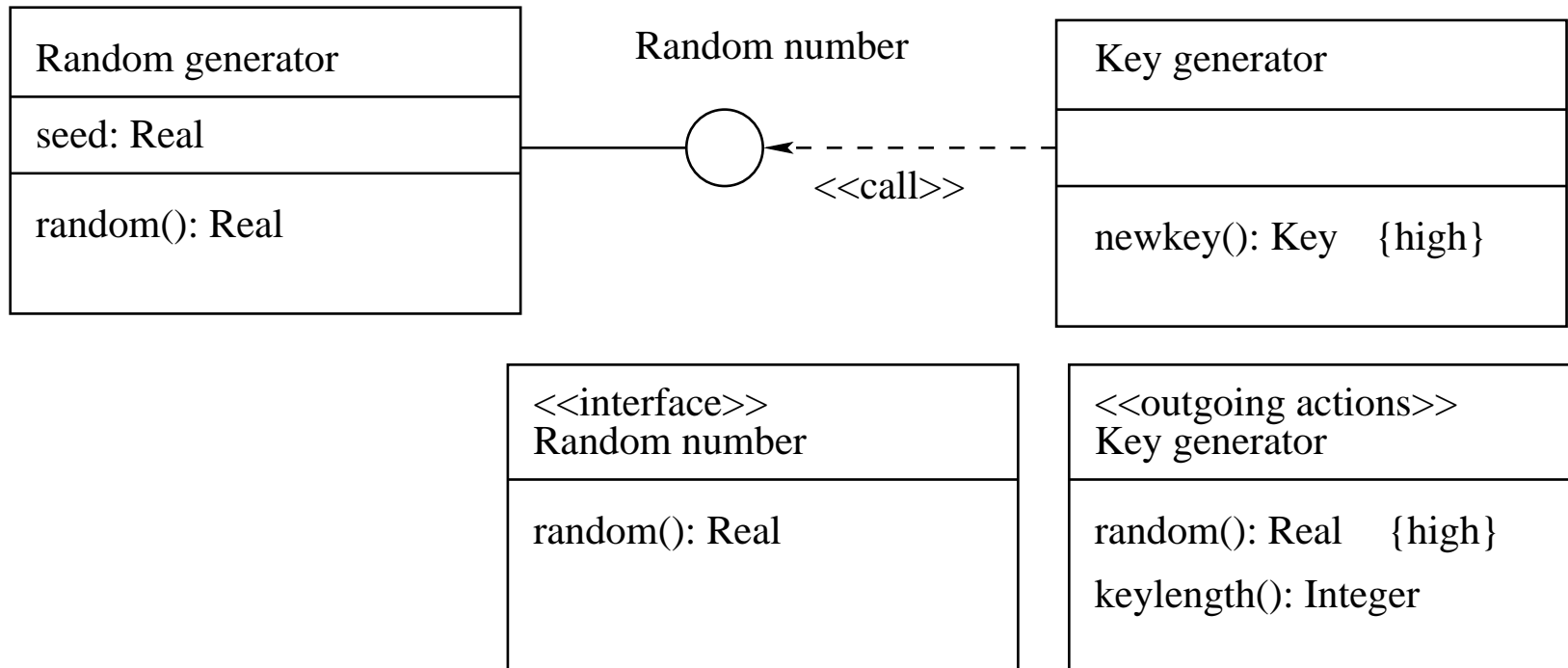
Class diagrams

Class diagrams: Classes and dependencies.

Dependency: Client, supplier, interface, stereotypes.

Class diagram is gives **secure dependency** if dependencies respect security levels.

Example: Key generator



random() does not provide required security level.

Distributed Objects

Objects distributed over **untrusted** networks.

“Adversary” intercepts, modifies, deletes, inserts messages.

Cryptographic protocols to exchange session keys etc.

Vulnerabilities often at **boundary** between protocols and system.

Expressions

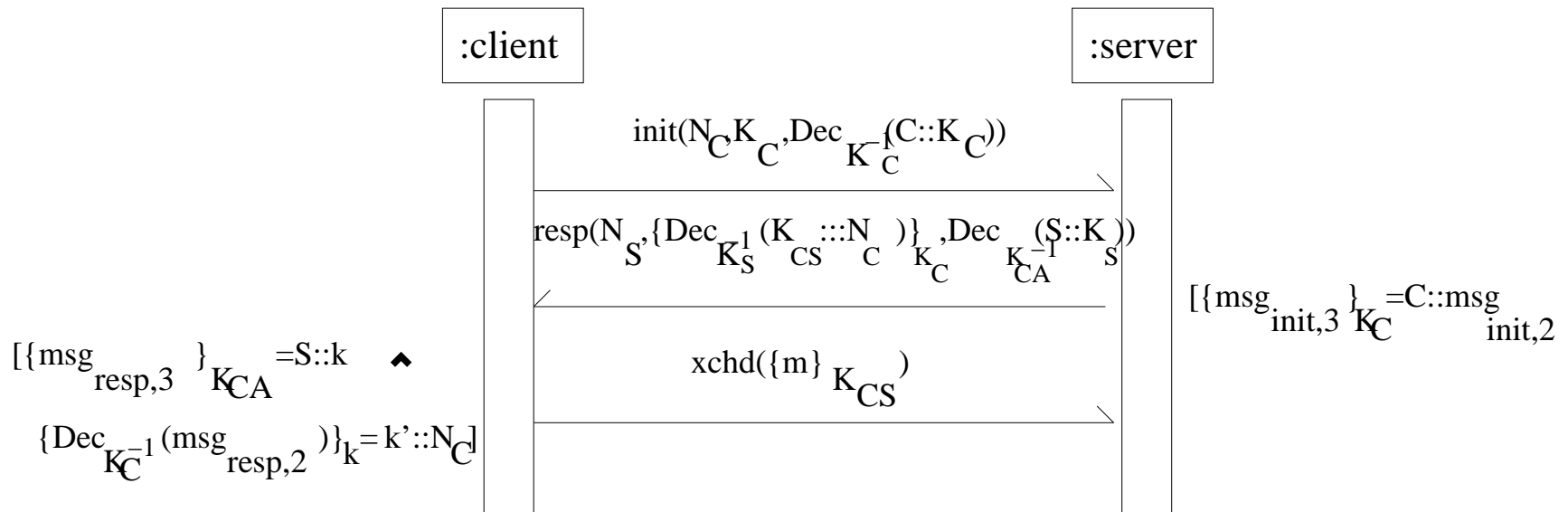
Exp: empty expression ε and the expressions:

$E ::=$	expression
d	$d \in \mathcal{D}$
K	key ($K \in \mathbf{Keys}$)
x	$x \in \mathbf{Var}$
$E_1 :: E_2$	concatenation
$\{E\}_e$	encryption ($e \in \mathbf{Keys} \cup \mathbf{Var}$)
$\mathcal{Dec}_e(E)$	decryption ($e \in \mathbf{Keys} \cup \mathbf{Var}$)

K^{-1} : decryption key corresponding to encryption key K .

Postulate $\mathcal{Dec}_{K^{-1}}(\{E\}_K) = E$.

Example: Proposed Variant of TLS (SSL)



Apostolopoulos, Peris, Saha; IEEE Infocom 1999

RSA encryption and signature (thus $\{\text{Dec}_{K^{-1}}(E)\}_K = E$).

Sequence diagrams: Secrecy

Sequence diagram defines stream-processing function f_P for each participant P .

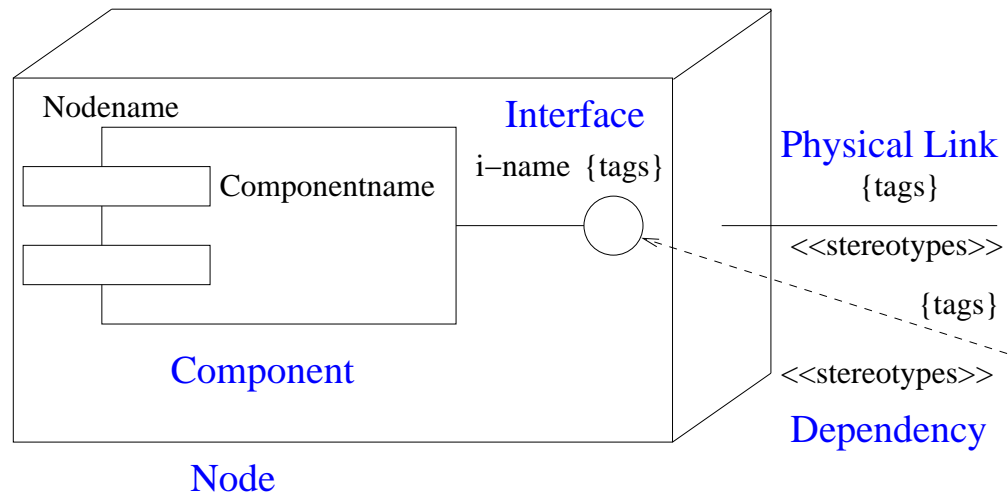
Additional adversary is modelled by function f_A .

The adversary **knows** $d \in \mathcal{D}$ if d appears in definition of f_A .

Sequence diagram S **preserves the secrecy** of d
if exists no adversary A (not knowing d)
such that composition of f_A with all f_P outputs d .

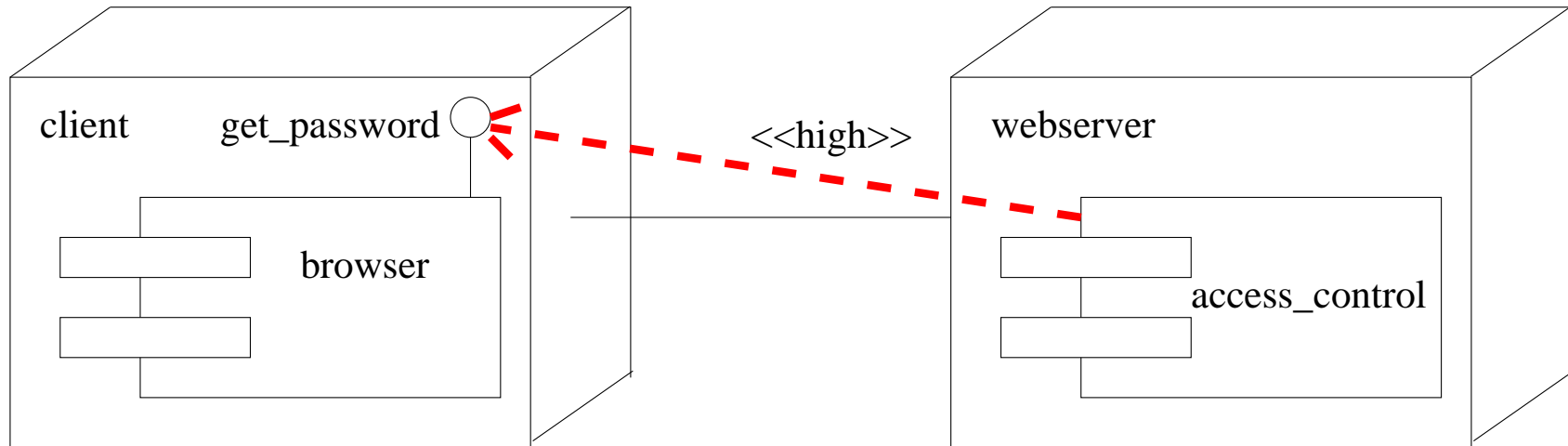
(Dolev, Yao 83)

Deployment diagrams: link security



Deployment diagram **provides communication security** if for each **high** dependency the physical connection provides **high** security.

Example



Physical connection does not provide the required security.

Related Work

Formal semantics for UML

Formal verification of security protocols

(Burrows, Abadi, Needham; Roscoe, Lowe, ...;
FME 01, MMM 01)

... and secure information flow (Goguen, Meseguer; Concur 00)

Security and software engineering (Devanbu, Fong, Stubblebine)

Conclusion

First step towards an extension of UML
for developing secure systems.

Further Work

Common Electronic Purse Specifications (Ifip SEC 01)

Encapsulating security engineering knowledge (IWSecP 01)

Compositionality, Refinement (AVoCS 01)

Design process

Java Security

Extension of UML using profiles

Future Work

More aspects of security.

Relate different views.

Tool support.