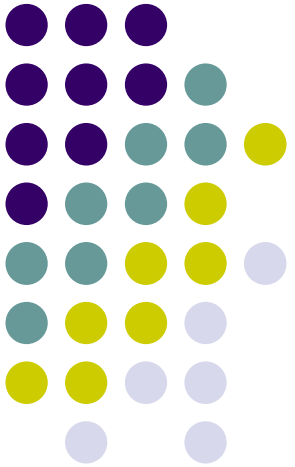
Security-Critical System Development with Extended Use Cases

Gerhard Popp
Jan Jürjens
Guido Wimmel

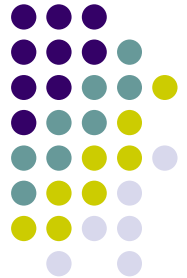
Munich University
of Technology
Germany

Ruth Breu

University of
Innsbruck
Austria

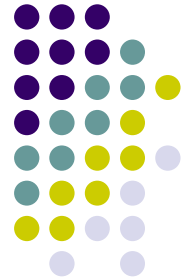


Overview



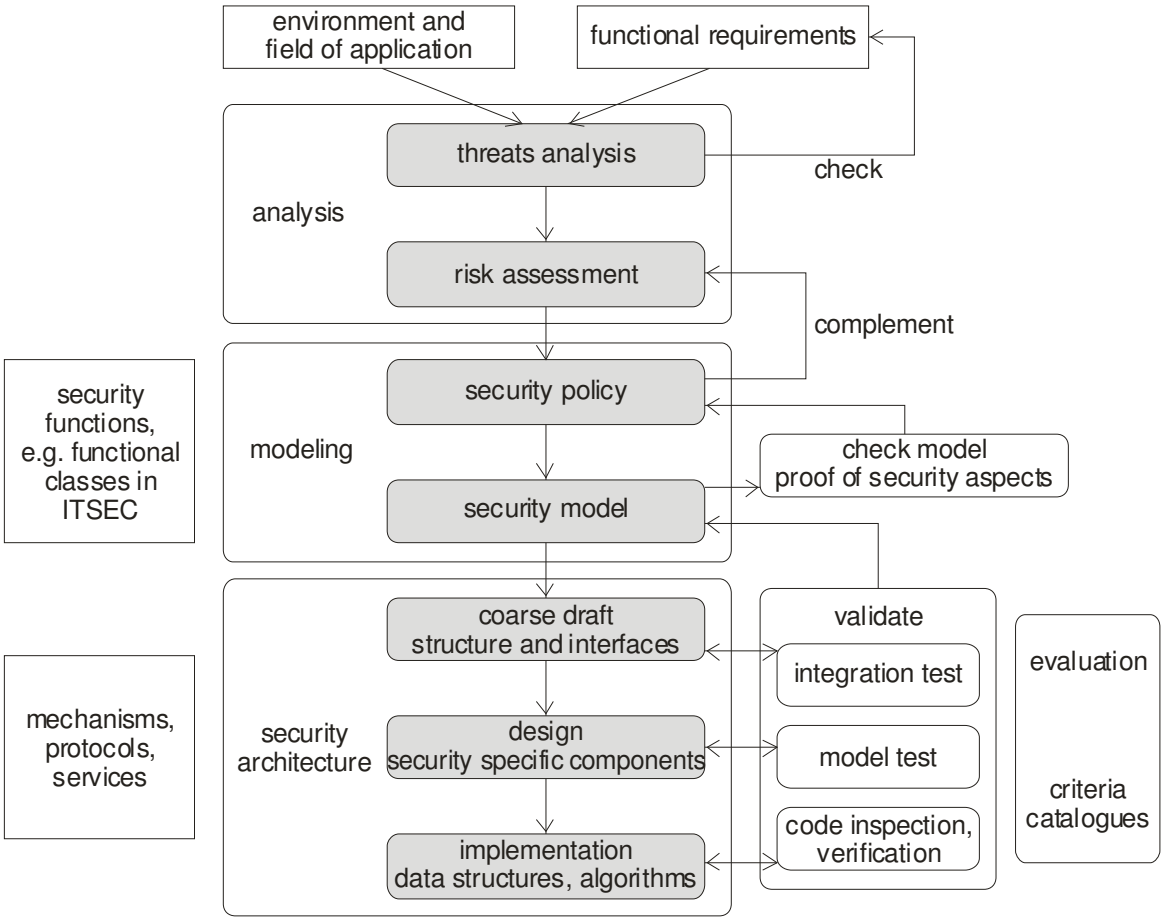
- Introduction
  - Critical System Development
  - Current Security Engineering Methodology
  - Use Case Development
- Security Use Case Development
  - Goals
  - Methodical Concept
  - Processing
  - Description Techniques
  - Static Security Data Modeling
  - Categorization
  - Integration into a development process
- Conclusion

Critical Systems Development



- High quality development of critical systems (dependable, security-critical, real-time, ...) is difficult.
- Many systems developed, fielded, used that do not satisfy their criticality requirements, sometimes with spectacular failures.
- Security is mostly an add-on to the common system development.

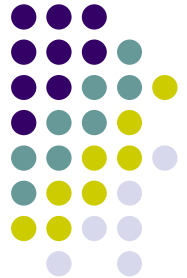
Current Security Engineering Methodology



What is missing?

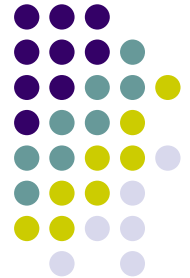
- common development of functionality and security
- no local specification of security
- product artifacts
- adaptable to other process models

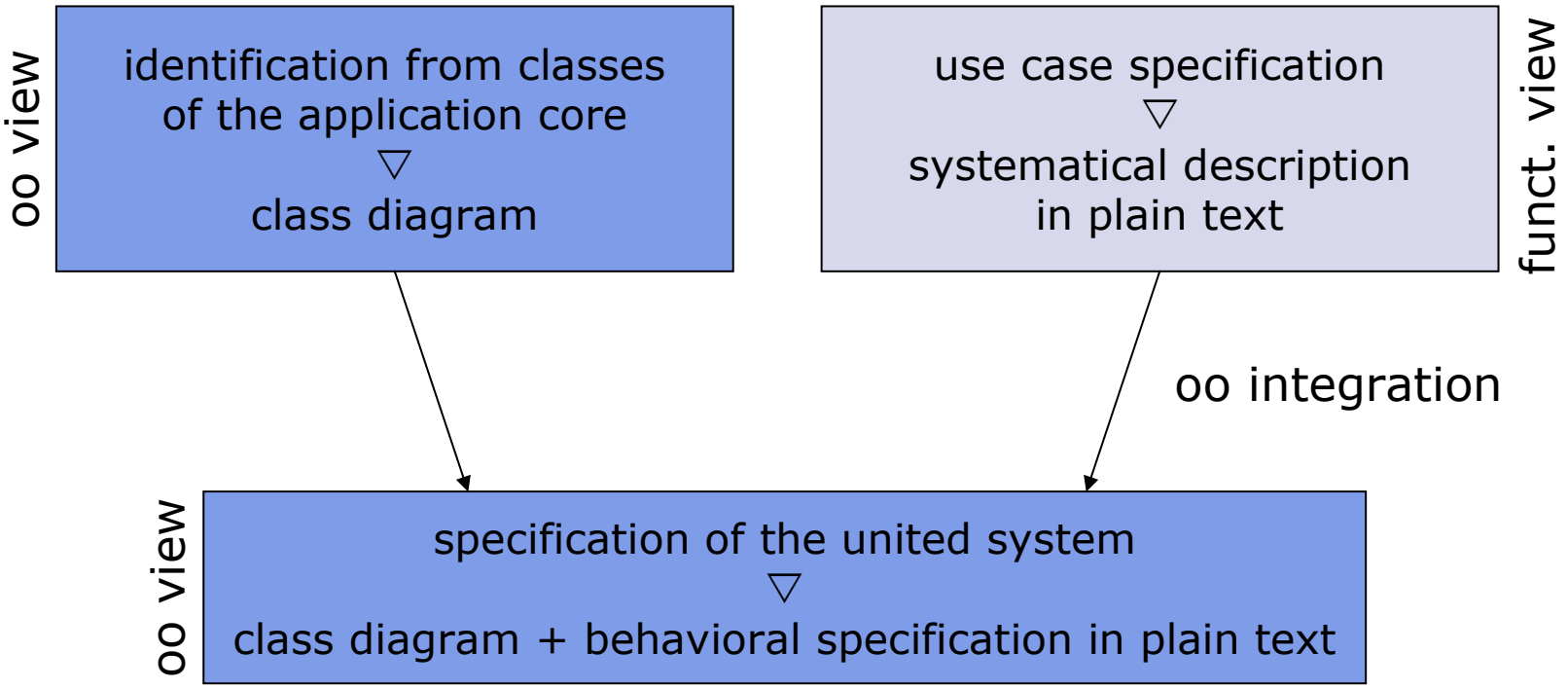
Use Case Oriented Development



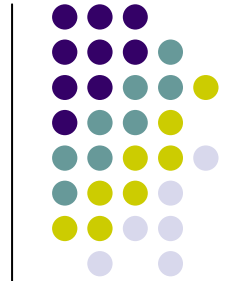
- Developers and customers think both about objects and about tasks in the early phases.
- Object-Oriented modeling of the data model.
- Use Cases for the modeling of dynamic aspects.

Methodical Concept for Use Case Oriented Development





Use Case and Class Diagram for Example Adjustment Posting

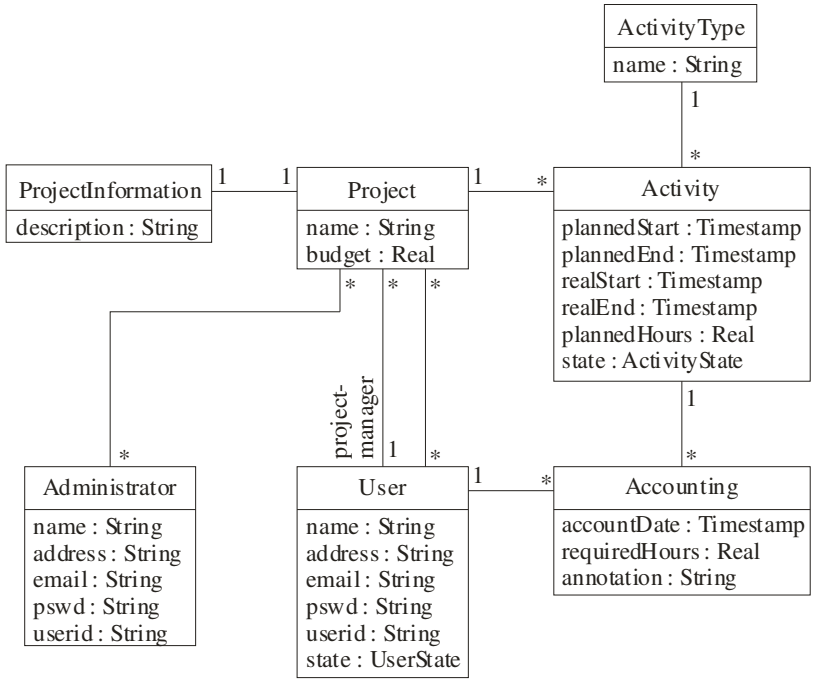


Actor: Team Worker
Input: AccountingID, Modified Accounting Data
Output: Accounting Data, Acknowledgement
Modified classes: Accounting
Description:

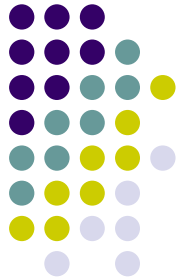
The system gets from the team worker an accounting identifier (e.g. from a prior search) and the system accounting manager searches the accounting and displays it to the team worker. He gives the modified data to the system and the system stores them. At the end, the system gives an acknowledgment back to the team worker.

Variants:

- The system cannot find the accounting from the accounting identifier. The system ends with a negative acknowledgement.
- The team worker gets the accounting and does not modify the data. The system hasn't to store the known data again, but the system returns a positive acknowledgement.
- The team worker puts in incorrect values. The system informs the user about the mistake and waits for a new input.

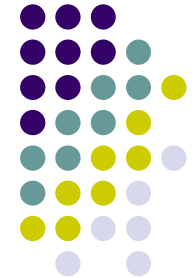


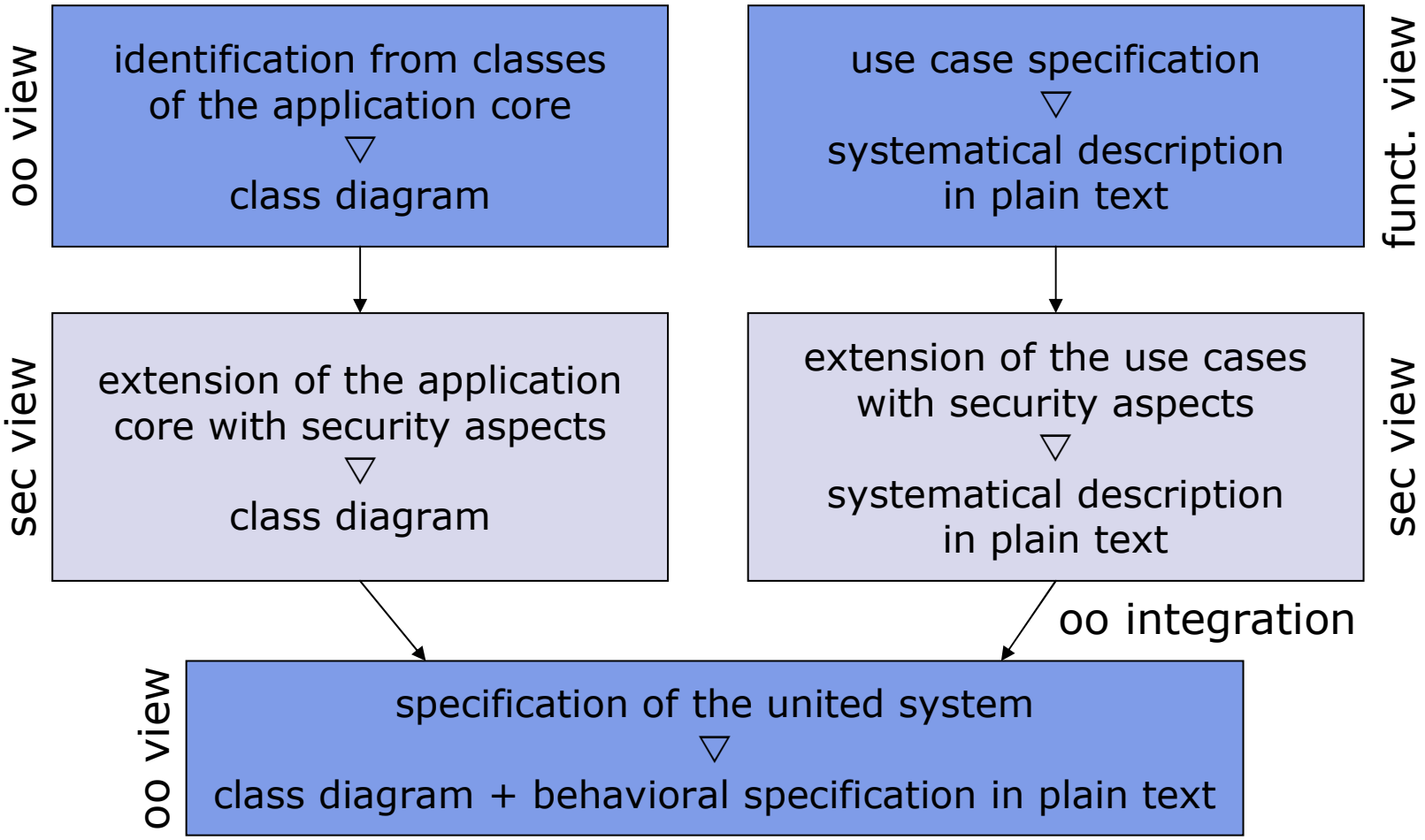
Security Use Case Goals



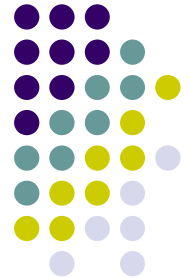
- Divide security in small, manageable pieces
- Continue security development in all phases, especially in the early phases
- Successive elaboration of security aspects
- Common development of functionality and security
- Local specification of security aspects
- Near the process also concentrate on artifacts
- Support iterative development, especially integration in an object oriented process

Methodical Concept for Security Use Case Oriented Development



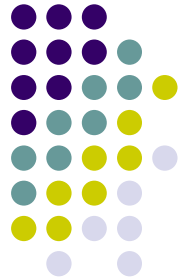


Processing Security Use Case Development



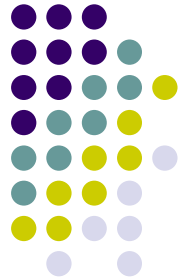
- Create use case as usual for the description of the functional behavior
- Check security objectives for the use case: authenticity, confidentiality, integrity, non-repudiation and availability
- Modeling threats for each threatened security objective
- Analyze the risk for each threat
- Design measures for each threat with potential risk
- Check correctness of the measures

Description Techniques for Security Use Cases



- Extended textual specification
  - Section Security
- Development of an access matrix
  - Local in the use case and integration in a global one
- Extended UML class diagrams for security aspects of the domain model

Example (1)



Create use case as usual

Check security objectives for the use case

…

Security

A1 (non-repudiation)
The adjustment posting has to be logged to the system

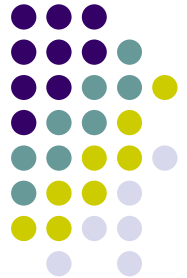
A2 (authenticity)
The team worker has to be authenticated before starting the use case

A3 (authenticity)
Web-Browser and TimeTool-System have to be authenticated before the transaction starts

A4 (confidentiality, integrity)
The use case must guarantee confidentiality and integrity

A5 (availability)
The use case must be available during extended working hours (6 a.m. to 22 a.m.)

Example (2)



Modeling threats and analyzing risks (here only for A1)

⇒ Non-repudiation for altering project activities is recognized

- Possible Threats
  - ProjectManager manipulates billable time
  - ProjectManager inputs negative time to manipulate budget overruns
- Risk Assessment
  - Probability is high, damage is substantial

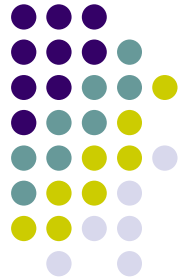
Measurements

- Introduce logging mechanism to store information before values are altered

Check

- Measure encounters problem: All changes and old values can be followed.
- In the activity involved people can look at the changes or could be informed about the alternation.

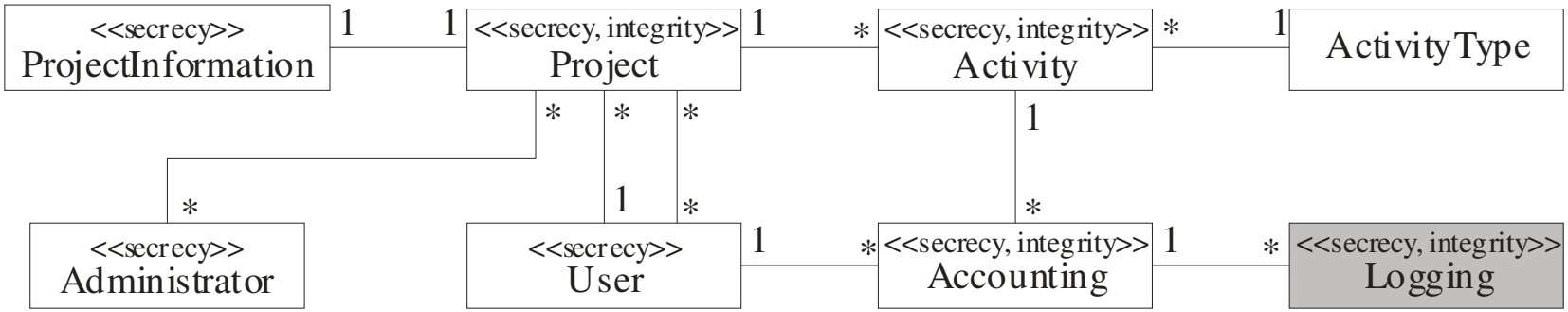
Static Security Data Modelling with UMLsec



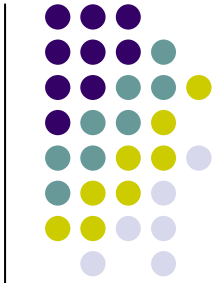
- Data modeling with class diagrams.
- Add security related information to class diagrams.
- Extension of class diagrams with:
  - Stereotypes << >>
  - Tags {tag=value}
  - Constraints

Example (3)

- Adding a class Logging to the class diagram
- **Iteration in the development and security process!**
- Behavior and Security in the use case is changing
- Security in the domain model is changing



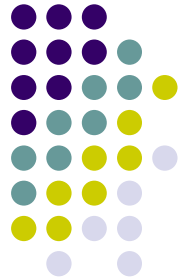
Example (4)



- Access matrix has to be extended and changed

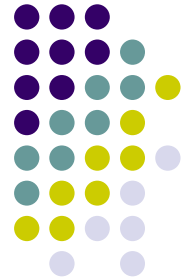
Actor \ Class	**Accounting**		**Logging**	
Team Worker	R	own accountings	R	all loggings to own accountings
	W	only after copying into logging	W	after creation
	C	-	C	for logging reasons
Projekt Manager	R	own accountings	R	all loggings to own accountings
	W	only after copying into logging	W	after creation
	C	-	C	for logging reasons
Administrator	R	all	R	all
	W	all	W	all
	C	all	C	all

Categorization of Security Use Cases

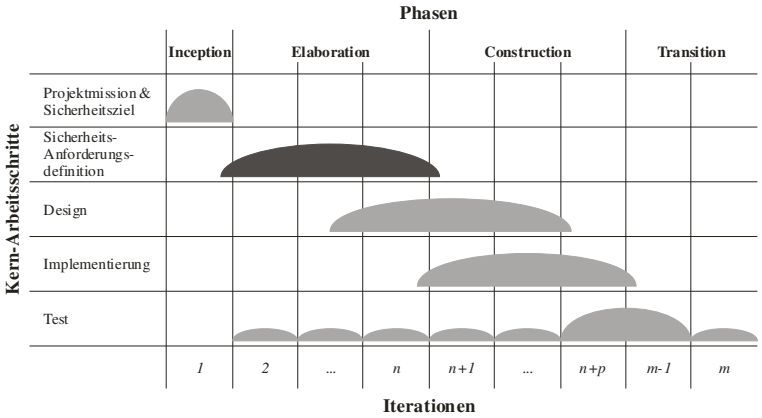


- 3 different types of security use cases
  - not security-critical
  - security critical
  - new use case for security aspects

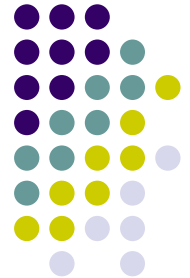
Integration of Security Use Cases in a Development Process



- Project Idea and Mission
- Business Process Engineering
  - Business behavior and processes
  - First class diagram
  - First access matrix
- Security Use Case Modeling
  - Security extended class diagram
  - Security extended use cases
  - Threats and Risk document
- Analysis
  - Sequence diagrams with security in the functional processing
- Design, Implementation, Test

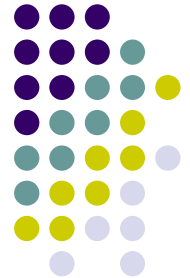


An Excursion to UMLsec



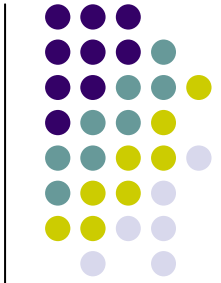
- UMLsec allows to express security-related information within UML diagrams
- Uses the UML extension mechanism
  - stereotypes
  - tagged values
  - constraints
- Book on related UMLsec
  Jan Jürjens: Secure Systems Development with UML, Springer-Verlag, 2004

Extension of Class Diagrams for Secure Data Modeling



Stereotype	Base Class	Tags	Description
secrecy	dependency		assumes secrecy
integrity	dependency		assumes integrity
high	dependency		high sensitivity
critical	object	secrecy integrity high	Critical object

Secure Data Modeling <<secure links>>

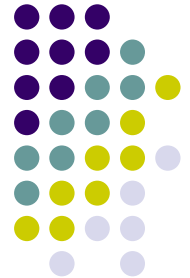


- Ensures that physical layer meets security requirements on communication.
- Constraint:
  For each dependency d with stereotype

  $$s \in \{\text{<<secrecy>>}, \text{<<integrity>>}, \text{<<high>>}\}$$

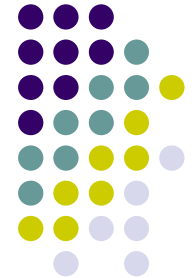
  between components on nodes $n \neq m$, have a communication link l between n and m with stereotype t such that:
  - if s = <<high>>: have $Threats_A(t) = \varnothing$.
  - if s = <<secrecy>>: have $read \notin Threats_A(t)$.
  - if s = <<integrity>>: have $insert \notin Threats_A(t)$.

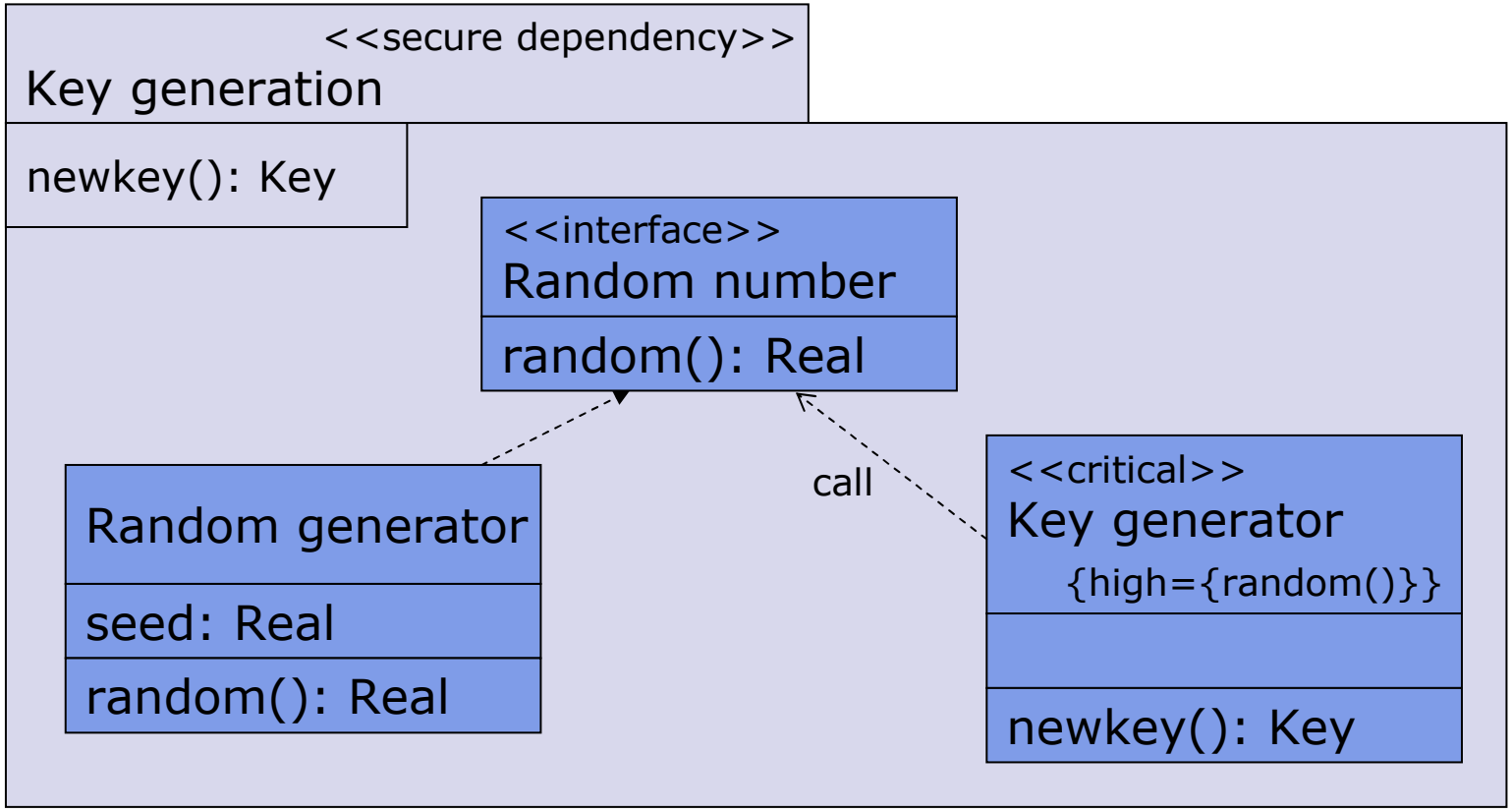
Secure Data Modeling <<secure dependency>>



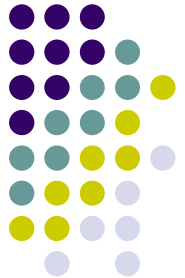
- Ensures that <<call>> and <<send>> dependencies between components respect security requirements on communicated data given by tags {secrecy}, {integrity}, {high}.
- Constraint:
  Given <<call>> or <<send>> dependency from C to D:
  - Any message n in D appears in {secrecy} in C if and only if does so in D.
  - If message in D appears in {secrecy} in C dependency stereotyped <<secrecy>>.
  - Analogously for {integrity} and {high}.

Key Generation Subsystem Instance (an Example)





The End



- Comments and Questions
- Thanks for your attention!