System-Level Design Under Confidentiality and Integrity Constraints

Janos Sztipanovits

VANDERBILT UNIVERSITY

Content

1. **Contextual Integrity**
2. Analysis of systems using formal models of privacy
3. Privacy aware system design
4. Example
   - CVRIA – Connected Vehicle Pilot
   - Analysis Architecture

Contextual Integrity

Helen Nissenbaum:

- "… technologies, systems, and practices that disturb our sense of privacy are those that have resulted in **inappropriate flows of personal information.**"
- "Inappropriate information flows are those that **violate context specific informational norms** … governing **respective social contexts.**"
- Parameters:
  - Actors: *sender, recipient, subject*
  - Information type: *context specific characterization of the content*
  - Transmission principle: *constraints over information flow*

Moving towards modeling

- Model disclosure, use of personal information
  - Messages has sender, receiver, subjects
- Privacy depends on context, sequence of actions
  - **Past and future relevant**
- Agents reason about attributes
  - **Deduction based on combining information**

Content

1. Contextual Integrity
2. Analysis of systems using formal models of privacy
3. Privacy aware system design
4. Example
   - CVRIA – Connected Vehicle Pilot
   - Analysis Architecture

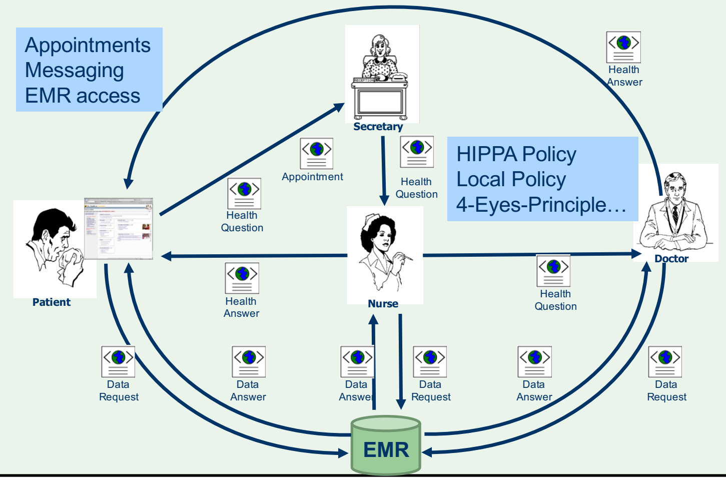
Logic of privacy and utility

- Agents in roles communicating attributes
- Communication action:
  - Sender (who sent the info)
  - Recipient (who received the info)
  - Subject (whom the info is about)
  - Attribute (type of info sent)
- Policy expressed in linear temporal logic over traces of communication
  - Temporal operators for "obligations"
- Example: MyHealthAtVanderbilt:

> In all states, only nurses and doctors receive health questions
>
> $G\ \forall p1, p2, q, m$
> $send(p1, p2, m) \land contains(m, q, health\text{-}question)$
> $\quad \rightarrow inrole(p2, nurse) \lor inrole(p2, doctor)$

Use of formal modeling

Workflows



Nurses should tag health questions

$G\ \forall p, q, s, m.\ inrole(p, nurse) \wedge send(p, q, m) \wedge contains(m, s, health\text{-}q) \rightarrow tagged(m, s, health\text{-}q)$

privacy

Doctors should answer health ques.

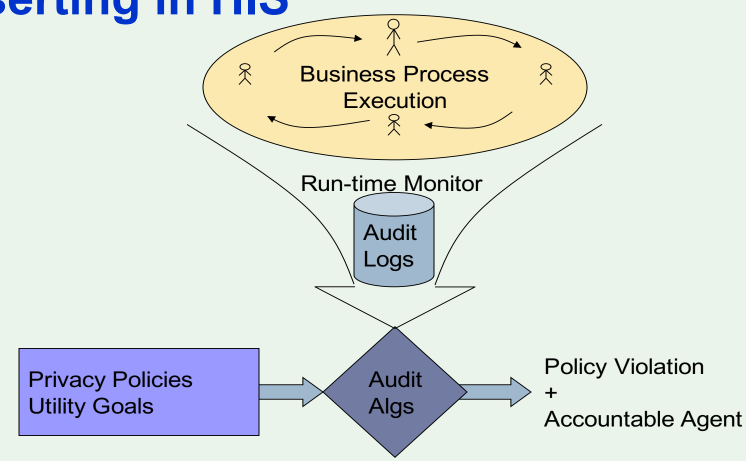
$G\ \forall p, q, s, m.\ inrole(p, doctor) \wedge send(q, p, m) \wedge contains(m, s, health\text{-}question) \rightarrow F\ \exists m'.\ send(p, s, m') \wedge contains(m', s, health\text{-}answer)$

utility

Sample Results

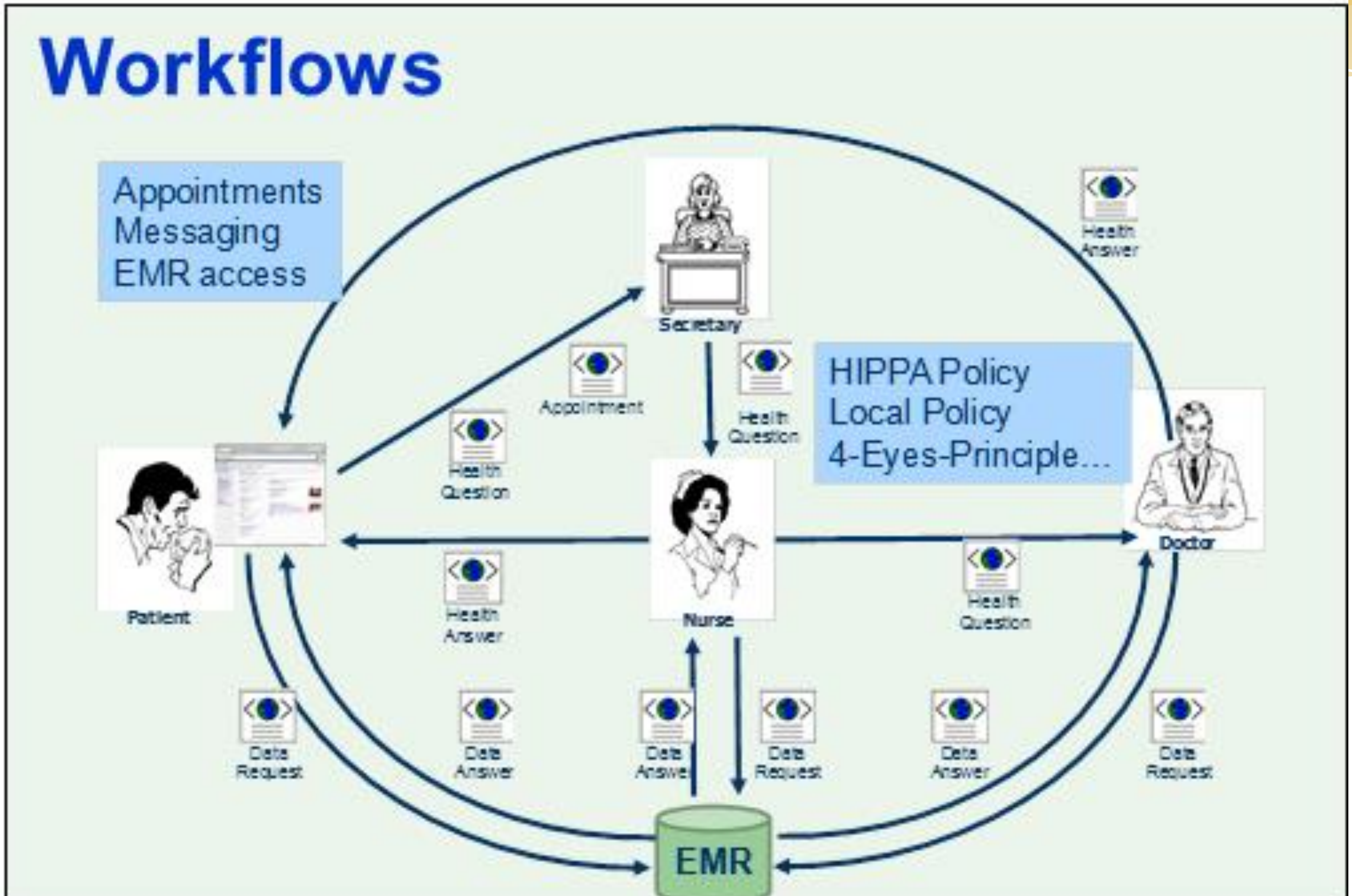
- If agents act responsibly, workflow achieves
  - Privacy in PSPACE: LTL model checking (SPIN)
  - Utility decidable: ATL* model checking (Mocha)
- Auditing
  - Find agents accountable for locally-compliant policy violation in graph-based workflows
  - Find agents who act irresponsibly using audit log
  - Algorithms use oracle: ***O***(msg) = contents(msg)
  - Minimize number of oracle calls

Inserting in HIS



auditing

Workflows-information flows- constraints



Policy language

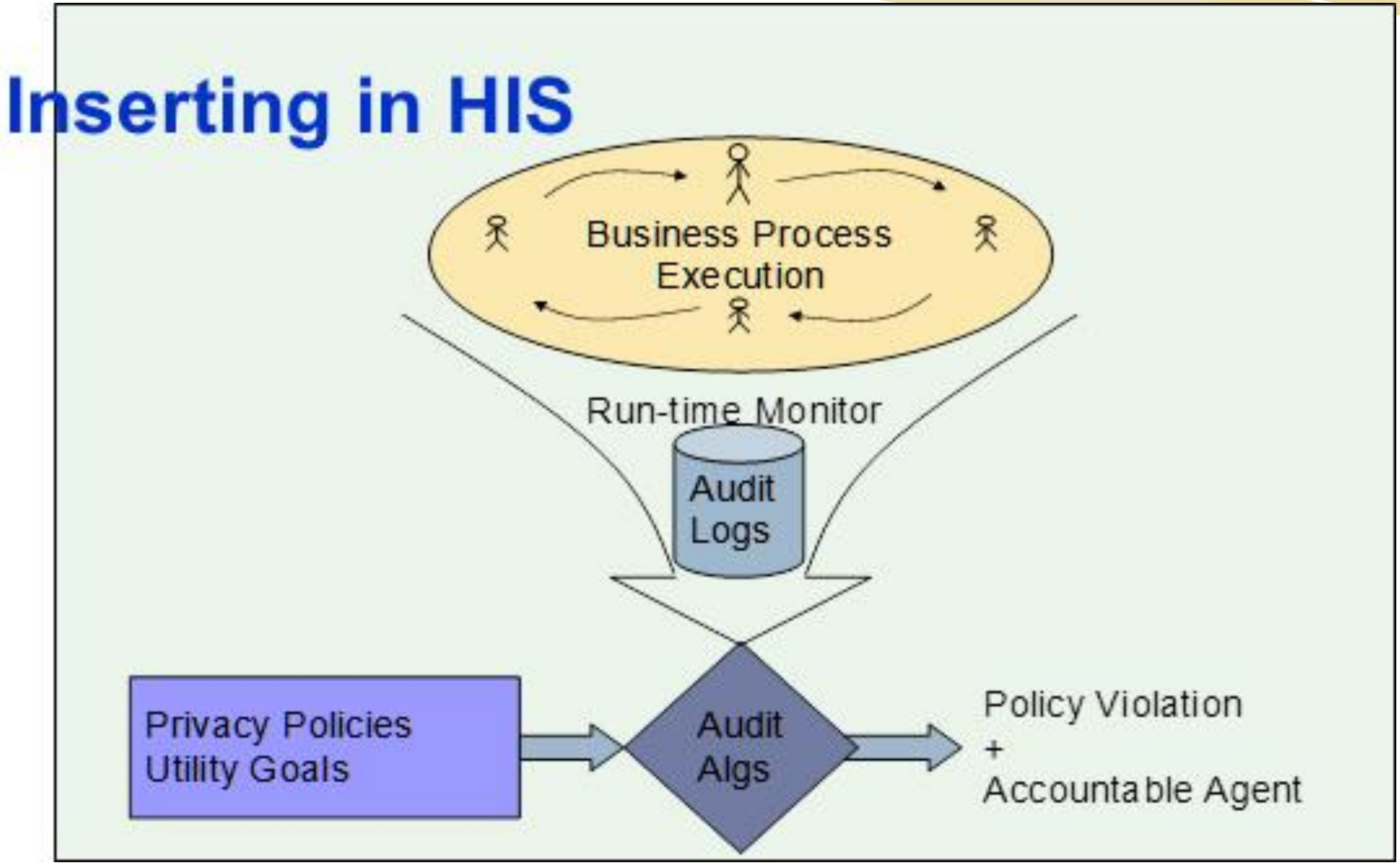
Nurses should tag health questions

$$\mathbf{G}\ \forall p, q, s, m.\ \text{inrole}(p, \text{nurse}) \wedge \text{send}(p, q, m) \wedge \text{contains}(m, s, \text{health-q}) \Rightarrow \text{tagged}(m, s, \text{health-q})$$

Doctors should answer health ques.

$$\mathbf{G}\ \forall p, q, s, m.\ \text{inrole}(p, \text{doctor}) \wedge \text{send}(q, p, m) \wedge \text{contains}(m, s, \text{health-question}) \Rightarrow \mathbf{F}\ \exists m'.\ \text{send}(p, s, m') \wedge \text{contains}(m', s, \text{health-answer})$$

Auditing


Inserting in HIS
Business Process
Execution
Run-time Monitor
Audit
Logs
Privacy Policies
Utility Goals
Audit
Algs
Policy Violation
+
Accountable Agent

Logic of privacy and utility: Logic Frameworks

Metric Linear Temporal Logic (MLTL)

Example-1: Lab result of any patient must be reported within 5 days when the measurement exceeds a given threshold (abnormal value)

$\forall p. \forall l. \forall m. lab(p,l,m) \land th<m \rightarrow \Diamond_{[0,6)} report(l)$

Example-2 Each lab result of a patient, who has a abnormal value in the last 30 days must be reported as suspicious within 2 days

$\forall p. \forall l. \forall m. lab(p,l,m) \land (\blacklozenge_{[0,31)} \exists l'. \exists m'. lab(p,l',m') \land \Diamond_{[0,6)} report(l')) \rightarrow \Diamond_{[0,3)} report(l)$

Linear Temporal Logic (LTL)

Example-1: Nurses should tag health questions

G $\forall$p, q, s, m. inrole(p, nurse) $\land$ send(p, q, m) $\land$ contains(m, s, health-question) $\rightarrow$ tagged(m, s, health-question)

Example-2: Doctors should answer health questions

G $\forall$p, q, s, m. inrole(p, doctor) $\land$ send(q, p, m) $\land$ contains(m, s, health-question) $\rightarrow$ **F** $\exists$m'. send(p, s, m') $\land$ contains(m', s, health-answer)

Goals in health care

Obtain a general framework for optimizing sequences of actions **under utility and privacy constraints** to address privacy challenges in patient portals and patient management systems.

- Formal language for privacy policies
  - Information organized by type
  - Policies describe permitted communication
  - Considers past and future communications
- Expresses much privacy legislation
  - HIPAA (Health Insurance Portability and Accountability Act)
  - COPPA (Children Online Privacy Protection Act)
  - GLBA (Gramm-Leach-Bliley Act, financial privacy)
- Analyze system designs for compliance

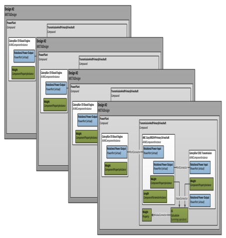
Content

1. Contextual Integrity
2. Analysis of systems using formal models of privacy
3. Privacy aware system design
4. Example
   - CVRIA – Connected Vehicle Pilot
   - Analysis Architecture

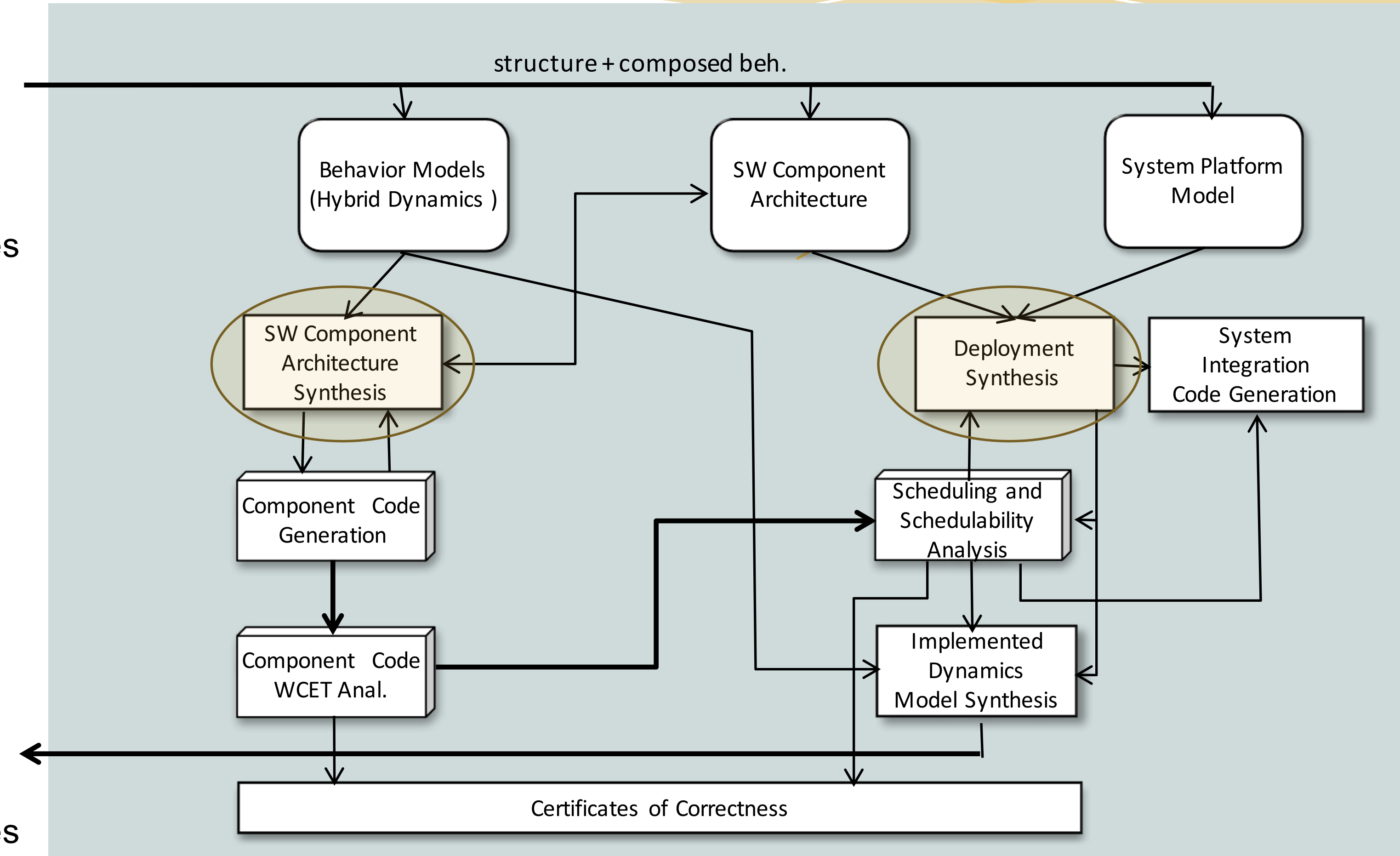
Privacy-aware System Design

- **The system-level synthesis problem for the "cyber" side of CPS:**
  - Derive specification for the behavior of the system components that will be implemented using networked computing
  - Derive a functional model for the information architecture and componentize the system
  - Select computing/networking platform
  - Derive deployment model assigning components of the information architecture to processing and communication platforms
  - Generate code for software components and derive WCET and WCCT
  - Perform timing analysis
- **Making security part of system-level co-design (correct-by-construction)**
  - Co-design of functionality, performance, timing and security
  - Our goal is to address security requirements as part of the design trades embedded in the system-level design process

Typical System-level Synthesis Steps of Information Architecture



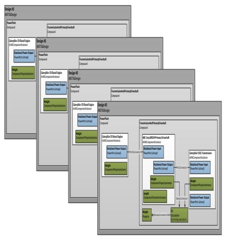
Design Architectures with ideal comp. dynamics



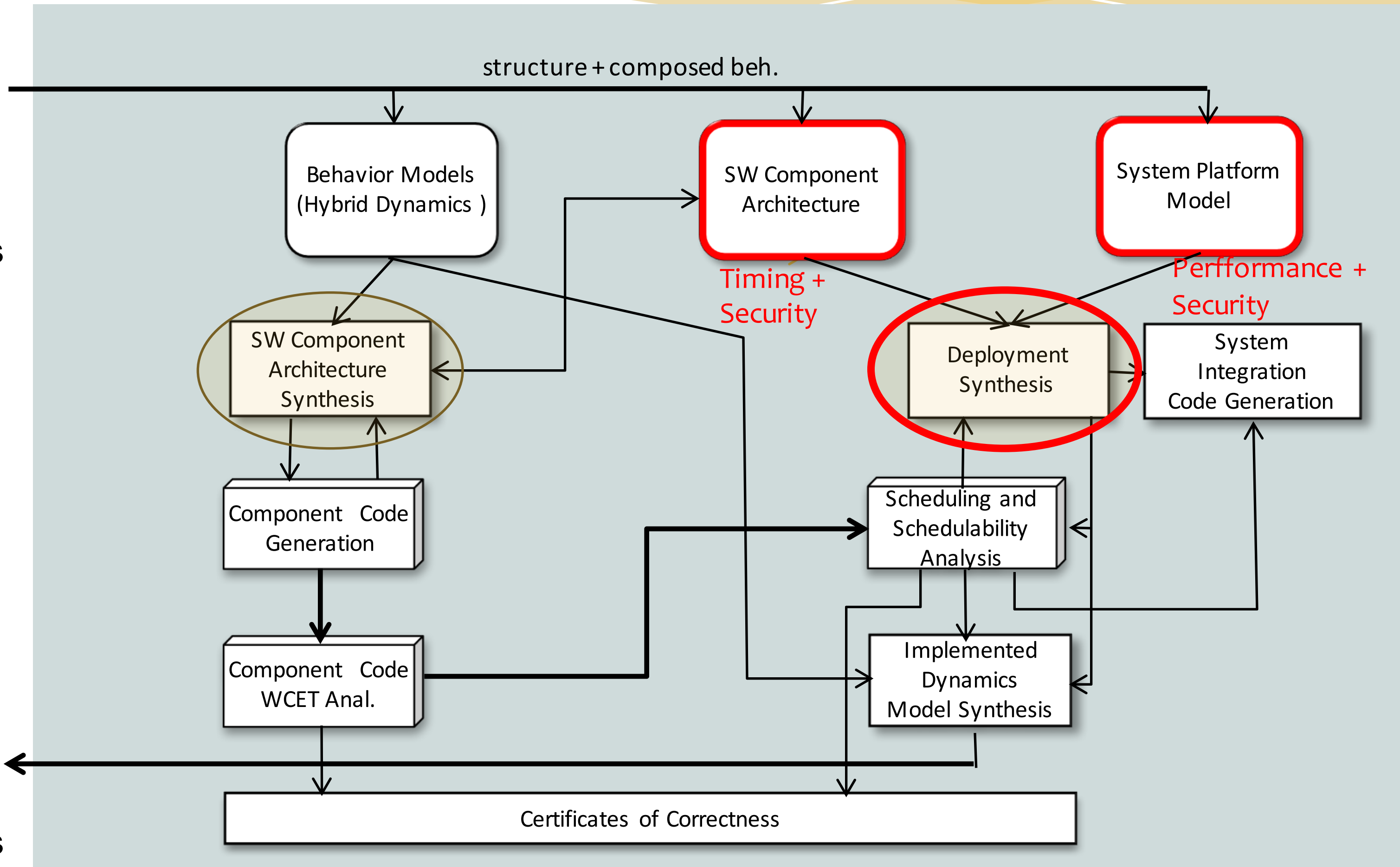
Design Architectures with **deployed comp. dynamics**

- CAN Bus
- TT bus

Typical System-level Synthesis Steps of Information Architecture



Design Architectures with ideal comp. dynamics



Design Architectures with **deployed comp. dynamics**

- CAN Bus
- TT bus

Synthesis Problem

- How to map a logical Information Architecture (components + information flows) on a physical Platform Architecture such that

  - Functional requirements (the information architecture)
  - Performance requirements (timing)
  - Security requirements (confidentiality and integrity)

  are satisfied simultaneously?

Challenges

- Modeling language suite ✓
  (behavior, information flows, SW components, architecture, timing, platform, deployment) - reuse previous work as example
- Security Requirement Modeling ✓
  (need to be composable with other modeling aspects)
- Common Semantic Domain and Formal Framework ✓
  (functional, performance and security models need to be anchored to a semantic domain suitable for synthesis)
- Synthesis Framework and Co-design flow ✓
  (mapping system-level synthesis problem on the formal framework and tools)
- Integrated Tool Suite and Validation
  (target domain rich enough for testing the co-design tool suite)

Content

1. Contextual Integrity
2. Analysis of systems using formal models of privacy
3. Privacy aware system design
   - **Decentralized Label Model**
   - Security types
   - Formal Framework
4. Example
   - CVRIA – Connected Vehicle Pilot
   - Analysis Architecture

Security Concerns Addressed

- **Integrity attacks**
  - Manipulate data (value, timestamp, source identity,..)
- **Confidentiality attack**
  - Leak critical data to unauthorized persons/systems
- **Integrity and confidentiality restrictions impose constraints on information flows.**
  - How to model these restrictions?
  - How to integrate these restrictions with others (functional and timing) and formulate a co-design problem?

Decentralized Label Model (DLM) for Information Flow Control

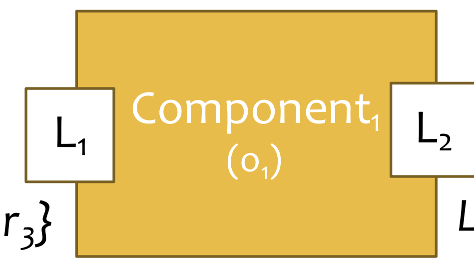
- Myers, Liskov (1997): Introduced **security-typed languages** by labeling variables with information flow security policies
- Method was developed for programming languages, the result is *Jif, a security-typed version of Java.*
- DLM provides mechanism for static/dynamic type checking of security labels in information flows to detect policy violations.
- Example: *Jif*, a security-typed version of Java
- **Introduce security-types in modeling languages**

DLM Concepts

- New semantic concepts introduced:
  - *Principles* that represent authority entities.
  - *Labels expressing* security classes encountered in most information flow models.
  - *Policies* that are elementary security primitives used in *labels.*
  - *Labeled entities* that have attached labels, such as *values, slots (variables, objects, i/o channels).* Copies of *values* can be relabeled, *slots* cannot.
  - *Operators* that can *relabel* or *declassify* values in information flows.
- The model can be naturally applied to system-level information flow modeling languages by assigning security types to input/output ports

Working With Security Labels

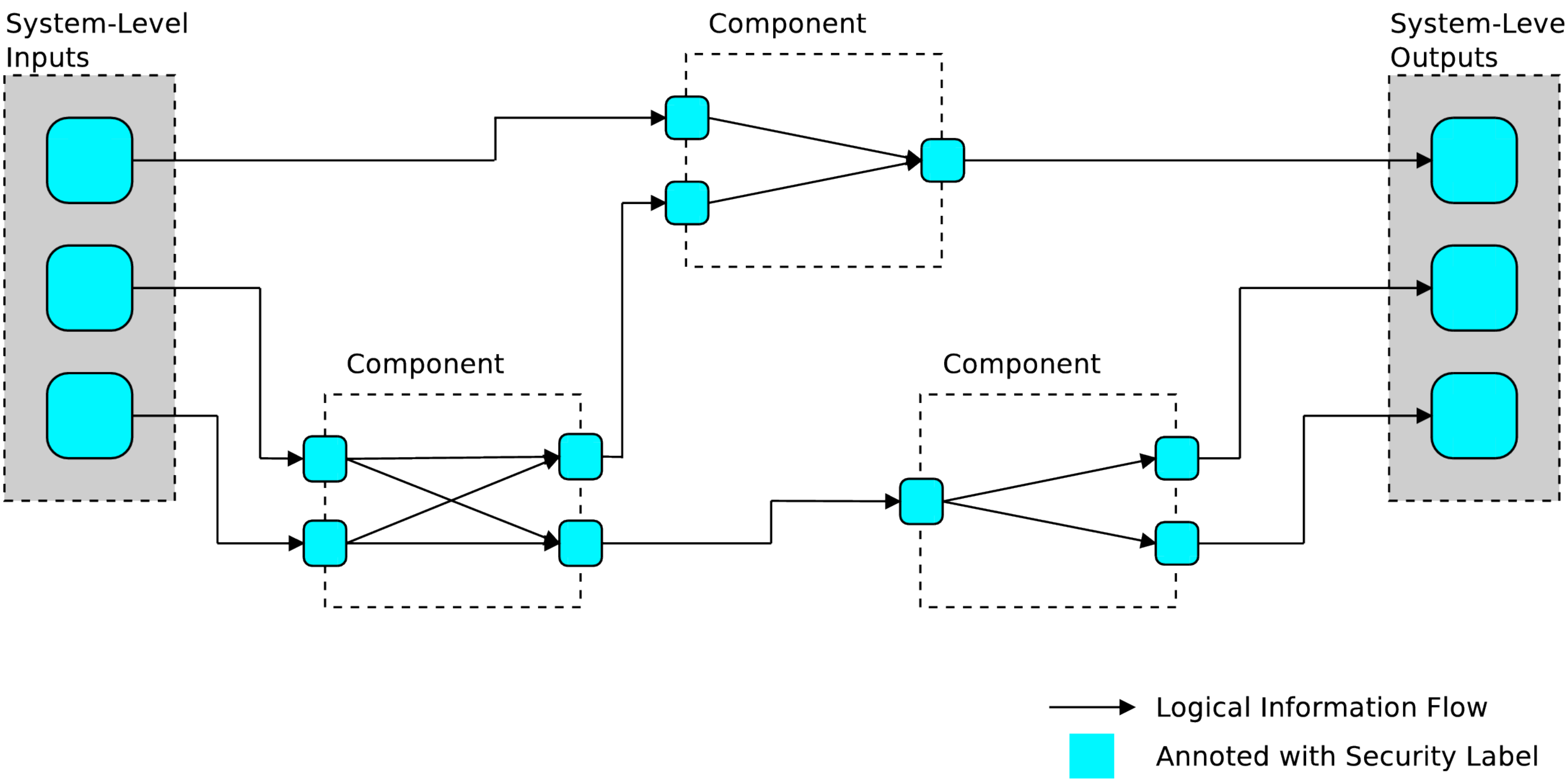
- *Labels* contain a set of *policies*. Each *policy* includes an *owner* and a set of *readers* allowed by the *owner*. The *effective reader set* for a *label* is the intersection of every *reader set* in it.
  $L = \{o_1: r_1, r_2; o_2: r_2, r_3\}$
- *Processing blocks* running under the authority of an *owner* can *declassify* the *owner's policy* by adding *readers.*



$L_1 = \{o_1: r_1, r_2; o_2: r_2, r_3\}$

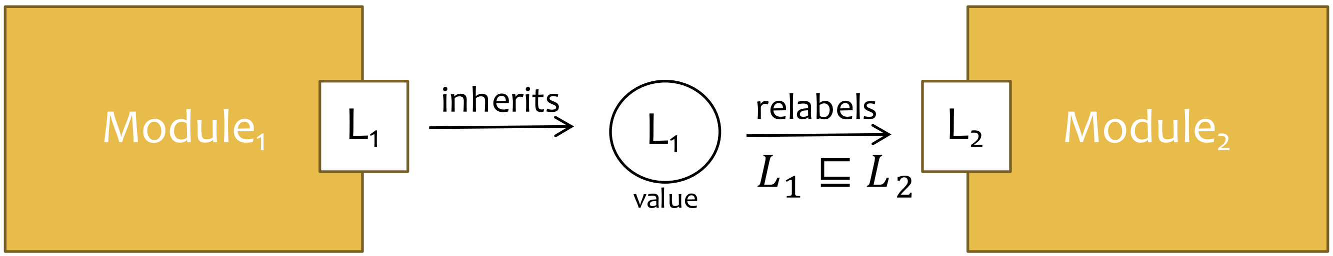
$L_2 = \{o_1: r_1, r_2, r_3, ; o_2: r_2, r_3\}$

Information Flows in an Information Architecture



Security Type Propagation Rules

- *Propagation rule-1 (restriction):*

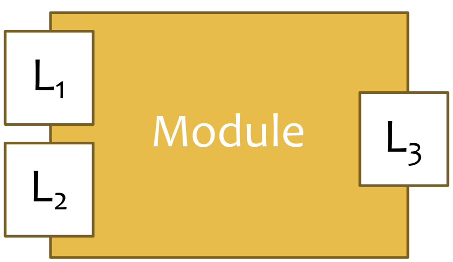


$owners(L_1) \subseteq owners(L_2)$

$\forall o \in owners(L_1), readers(L_1, o) \supseteq readers(L_2, o)$

(L_1 has more readers and fewer owners than L_2)

- *Propagation rule-2 (join):*



L_3 is the *join* of L_1 and L_2

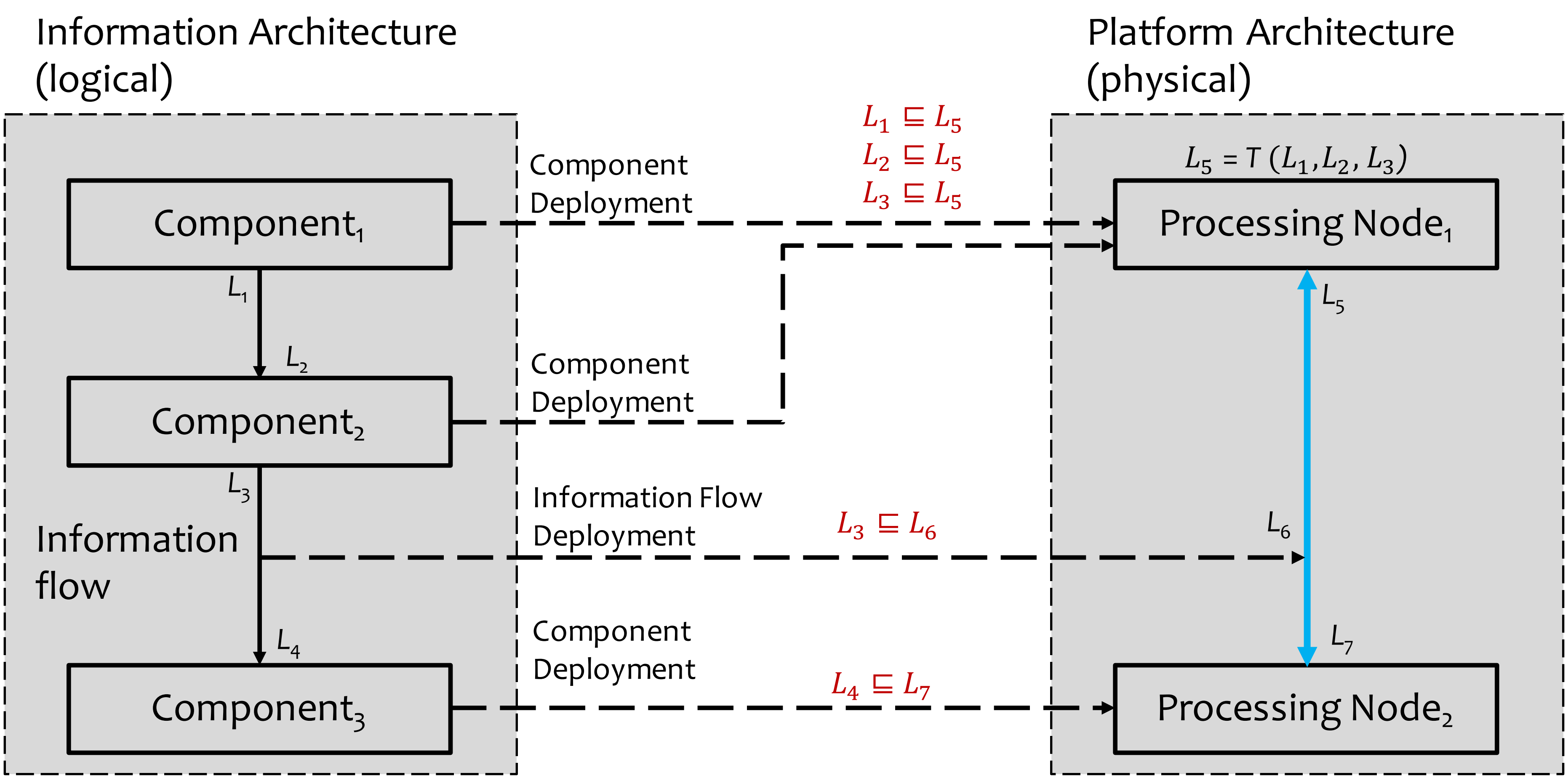
$L_3 = L_1 \sqcup L_2$

$owners(L_1 \sqcup L_2) = owners(L_1) \cup owners(L_2)$

$readers(L_1 \sqcup L_2, o) = readers(L_1, o) \cap readers(L_2, o)$

(*join* L_1 and L_2 is the least restrictive label that maintains all the flow restrictions specified by L_1 and L_2)

Information Architecture Deployed on a Physical Platform



Content

1. Contextual Integrity
2. Analysis of systems using formal models of privacy
3. Privacy aware system design
4. Example
   - CVRIA – Connected Vehicle Pilot
   - Analysis Architecture

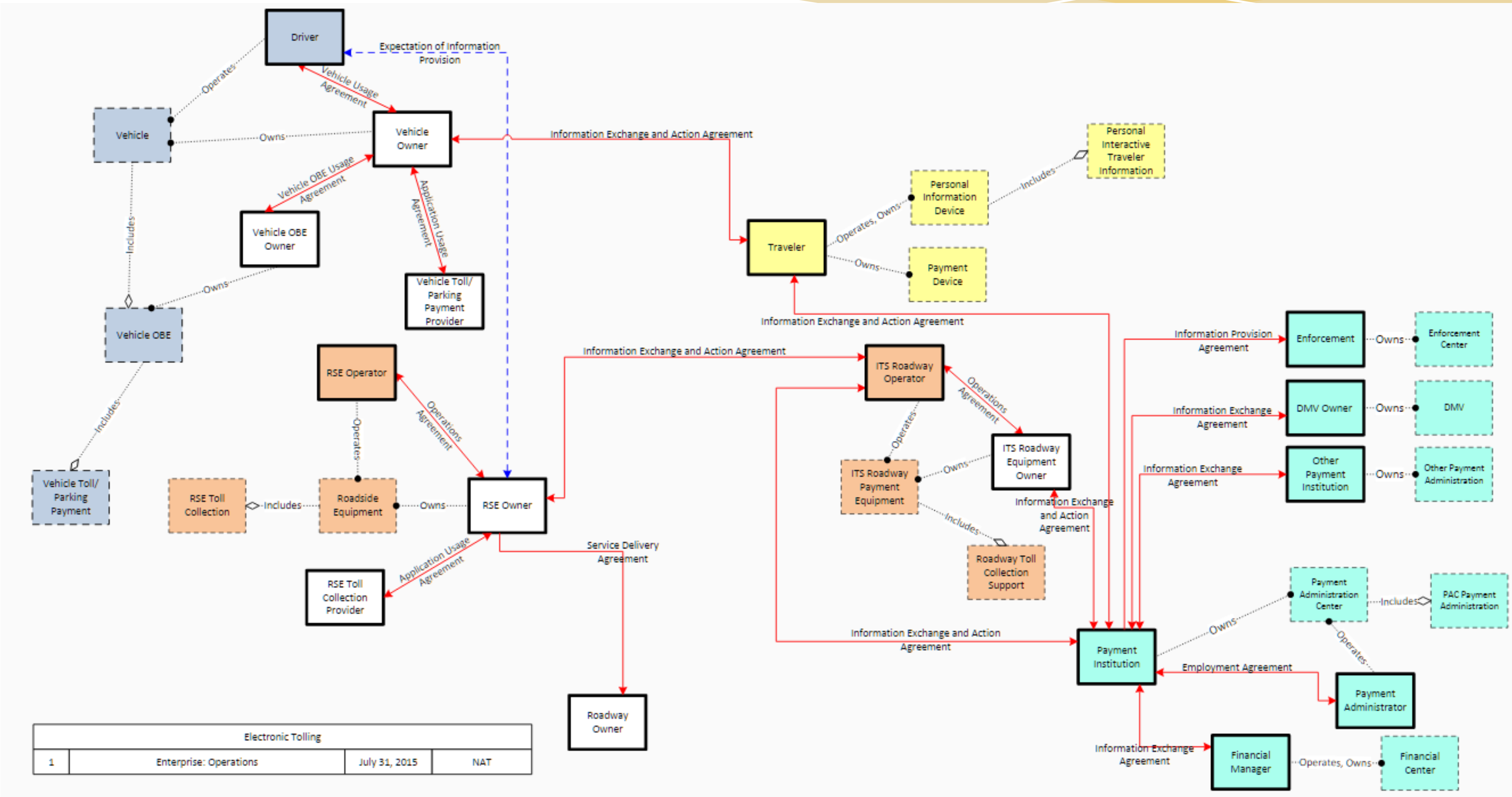
Example: CVRIA Models

- It provides a large set of use cases and architecture models. In general the models contain:
  - Organizational, physical, functional, and communication models (e.g.: toll collection, vehicles, drivers, equipment)
  - Dataflows, information flows, data structures between components
  - Mappings across different layers
- Validation concept
  - Import relevant subset of CVRIA models into our formal modeling environment (FORMULA)
  - Integrate CVRIA DSMLs with security DSL constructs
  - Based on the security labels:
    - Perform type checking (using FORMULA constraint check)
    - Propagate security labels (using FORMULA)
    - Find deployment properties (using FORMULA Z3 solver)

Example Application: Electronic Toll Collection

- Collect tolls electronically
- Detect and process violations
- Fees may be adjusted to implement demand management strategies
- Communication between roadway equipment and the vehicle is required
- Fixed-Point to Fixed-Point interfaces between toll collection equipment and transportation authorities and financial infrastructure supporting fee collection
- Toll violations are identified and electronically posted

Enterprise View: Agents



Electronic Toll Collection: Functional View

Processes: 48

Level	Name	Type	Allocated to Application Object
5.4	Provide Law Enforcement Allocation	Collection	
5.4.2	Process Violations for Tolls	Pspec	- PAC Payment Administration
6.7	Provide Driver Personal Services	Collection	
6.7.1	Provide On-line Vehicle Guidance	Collection	
6.7.1.2	Provide Driver Guidance Interface	Pspec	
6.7.3	Provide Traveler Services in Vehicle	Collection	
6.7.3.3	Provide Driver Information Interface	Pspec	
7.1	Provide Electronic Toll Payment	Collection	
7.1.1	Process Electronic Toll Payment	Collection	

7.4.1.2	Process Travel Services Provider Payments
7.4.1.3	Process Driver Map Update Payments
7.4.1.4	Process Traveler Map Update Payments
7.4.1.5	Process Traveler Other Services Payments
7.4.1.6	Process Traveler Trip and Other Services Payments

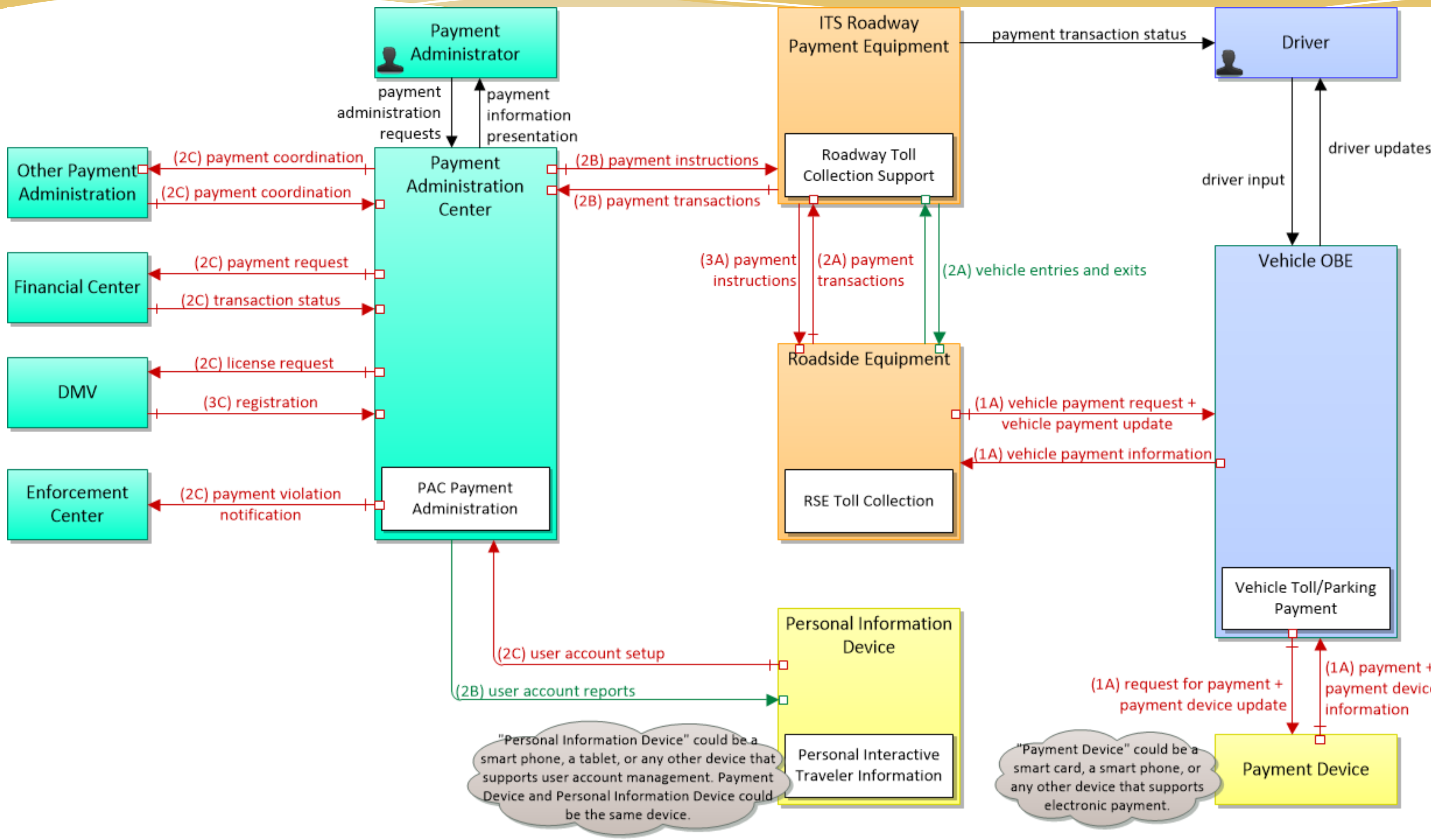
7.1.4	Provide Driver Toll Payment Interface	Pspec	
7.1.7	Provide Payment Device Interface for Tolls	Pspec	- Vehicle Toll/Parking Payment
7.1.8	Exchange Data with Other Payment Administration	Pspec	- PAC Payment Administration
7.2	Provide Electronic Parking Payment	Collection	
7.2.7	Provide Payment Device Interface for Parking	Pspec	
7.4	Carry-out Centralized Payments Processing	Collection	
7.4.1	Collect Advanced Payments	Collection	
7.4.1.8	Process Electric Charging Payments	Pspec	- PAC Payment Administration
7.4.1.9	Process Roadside Electric Charging Payments	Pspec	
7.4.1.10	Process Vehicle Electric Charging Payments	Pspec	
7.5.1	Provide Vehicle Payment Device Interface	Pspec	
7.5.3	Provide Personal Payment Device Interface	Pspec	- Personal Interactive Traveler Information
7.6.1	Process VMT Payment	Collection	
7.6.1.1	Collect Road Use Charging Data	Pspec	
7.6.1.3	Bill Driver for Road Use Charges	Pspec	- RSE Toll Collection
7.6.1.4	Manage Road Use Charging Price Data	Pspec	- PAC Payment Administration
7.6.1.5	Manage Road Use Charges Processing	Pspec	- PAC Payment Administration
7.6.2	Support Road Use Charging	Pspec	
7.6.3	Provide Driver Road Use Charging Payment Interface	Pspec	
7.6.4	Provide Payment Device Interface for Road Use Charging	Pspec	

Dataflows: 85

Source Pspec	Data Flow	Destination Pspec
Administer Multimodal Payments	multimodal_payment_request_to_field	Bill Driver for Road Use Charges
Administer Multimodal Payments	multimodal_toll_payment_data	Manage Toll Processing
Administer Multimodal Payments	traveler_personal_multimodal_payment_request	Provide Personal Payment Device Interface
Administer Multimodal Payments	traveler_personal_multimodal_account_reports	Provide Personal Payment Device Interface
Bill Driver for Road Use Charges	multimodal_payment_confirmation_from_field	Administer Multimodal Payments
Bill Driver for Road Use Charges	current_toll_transactions_from_roadside	Bill Driver for Tolls
Bill Driver for Road Use Charges	road_use_payment_collected_from_field	Manage Road Use Charges Processing
		Manage Road Use … Charging
Bill Driver for Road Use Charges	toll_vehicle_payment_data_request	Provide Payment Device Interface for Tolls
Bill Driver for Road Use Charges	toll_payment_debited	Provide Payment Device Interface for Tolls
Bill Driver for Road Use Charges	toll_payment_request	Provide Payment Device Interface for Tolls
Bill Driver for Road Use Charges	toll_vehicle_payment_data_clear	Provide Payment Device Interface for Tolls
Bill Driver for Road Use Charges	toll_vehicle_payment_data_update	Provide Payment Device Interface for Tolls
Bill Driver for Road Use Charges	toll_payments_from_roadside	Read Vehicle Payment Data for Tolls
Bill Driver for Tolls	toll_roadside_payment_billing	Bill Driver for Road Use Charges
Bill Driver for Tolls	toll_bad_payment_check_request	Manage Bad Toll Payment Data

Administer Multimodal Payments	multimodal_payment_request_to_field	Bill Driver for Road Use Charges
Administer Multimodal Payments	multimodal_toll_payment_data	Manage Toll Processing
Administer Multimodal Payments	traveler_personal_multimodal_payment_request	Provide Personal Payment Device Interface

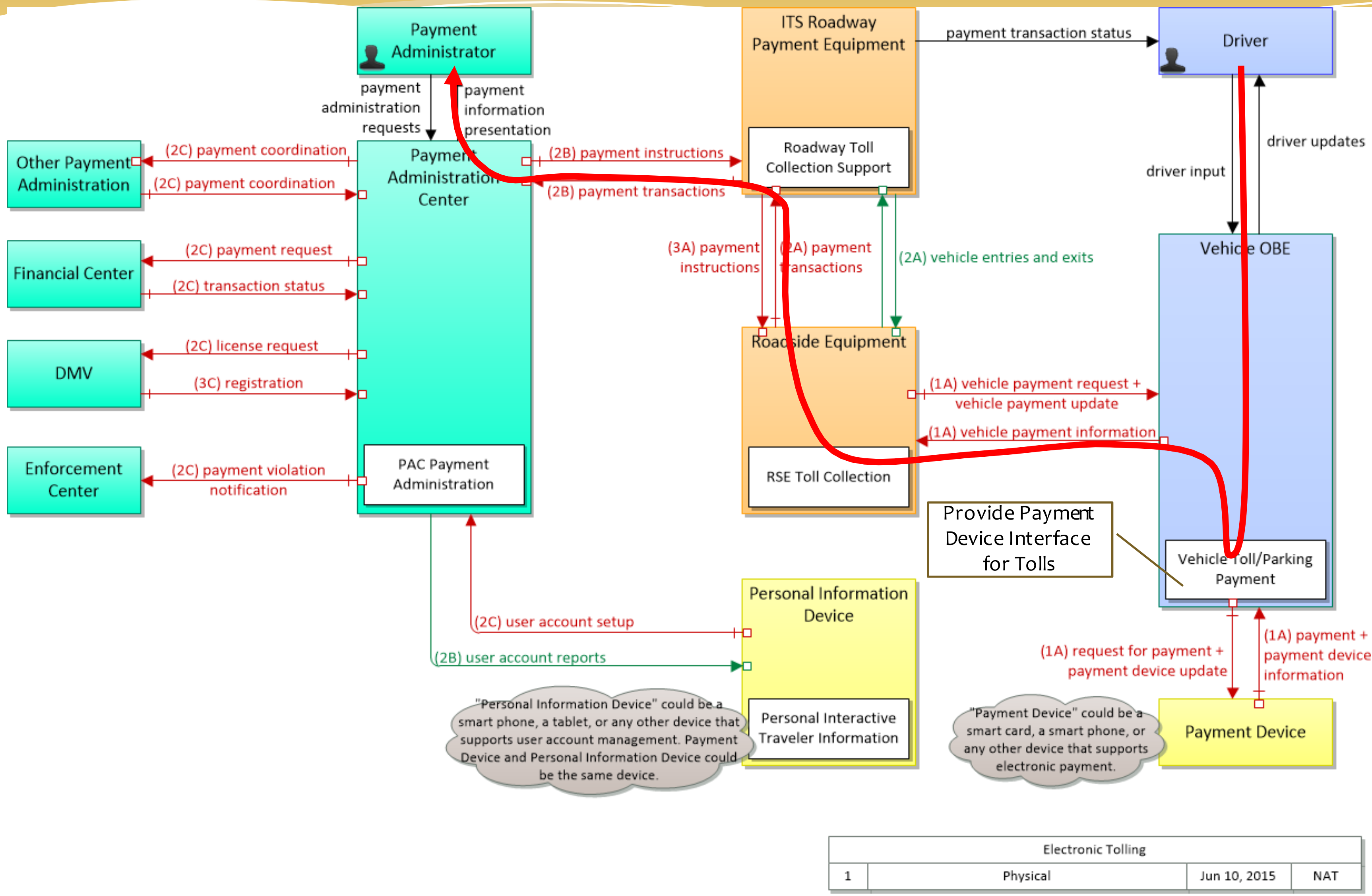
Electronic Toll Collection: Physical View Platform Architecture



http://www.iteris.com/cvria/html/applications/app109.html#tab-3

Electronic Tolling			
1	Physical	Jun 10, 2015	NAT

Dataflow Example



Electronic Toll Collection – A Glimps of Functional to Physical Deployment Mappings