Addressing Early Life Cycle Privacy Risk

Applying System-Theoretic Early Concept Analysis and Model-Based Systems Engineering to Privacy

Stuart S. Shapiro
The MITRE Corporation
Bedford, MA USA
sshapiro@mitre.org

Abstract—**This paper adapts System-Theoretic Early Concept Analysis (STECA), an instrumental safety risk management technique, for privacy to better identify and address privacy risks early in the engineering process. The technique, STECA-Priv, aims to infer a nominal functional privacy control structure based on a conceptual system description and privacy-related system behavioral constraints. Model-based systems engineering (MBSE) is employed in conjunction with STECA-Priv to validate the projected control structure and to identify privacy risks in the form of constraint violations. To illustrate STECA-Priv as supported by MBSE, it is applied to the simplified example of a smart television.**

Keywords—privacy risk; System-Theoretic Early Concept Analysis; STECA; STECA-Priv; model-based systems engineering; MBSE

I. Introduction

Engineering invariably involves both analytical and instrumental methods. The former tend to be more straightforward than the latter as they target an extant situation or specification, analyzing something that in some shape or form already exists. Instrumental methods, in contrast, support the creation of something new. It is not surprising, then, that methods for managing privacy risk as part of socio-technical system design are thinner on the ground than methods for assessing risk in already designed (and possibly already implemented) systems. This is not to imply the adequacy of current methods for privacy risk analysis, but rather to note that however problematic the state of analytical privacy risk management techniques, the state of instrumental privacy risk management techniques is even more so.

Previously [1], we sought to add to the stable of analytical privacy risk management techniques by adapting a methodology [2] initially developed to assess safety risk in support of safety engineering—System-Theoretic Process Analysis (STPA), grounded in System-Theoretic Accident Model and Processes (STAMP)—and later extended to assess security risk in support of security engineering [3]. Here, we aim to do the same for instrumental privacy risk management techniques by adapting another STAMP-related technique, System-Theoretic Early Concept Analysis (STECA), developed for use in the early stages the systems engineering life cycle (SELC) [4]. (Extensions of STECA supporting other types of engineering doubtless are possible, as has been the case with STPA.) The “analysis” in STECA is deceptive, though, as the target of that analysis is intended to be a concept of operations (ConOps), a natural language description of the system’s operation, which provides inputs to a more fundamentally instrumental process of postulating an appropriate control structure.

In addition to adapting STECA for privacy (yielding STECA-Priv) in much the same way we adapted STPA for privacy (yielding STPA-Priv), we leverage model-based systems engineering (MBSE) as a mechanism for checking the integrity of the postulated control structure. (STECA itself was partially inspired by MBSE [4].) In MBSE, complex systems are represented as models using diagrammatic modeling languages such as Unified Modeling Language (UML) or Systems Modeling Language (SysML) and current tools render these models executable. This presents opportunities for detection of unanticipated emergent properties and helps ensure consistent and up-to-date life cycle documentation, since the model is used to generate these documents. The International Council on Systems Engineering (INCOSE) and the Object Management Group (OMG) [5] are among the organizations driving the development of MBSE and it has been adopted for systems engineering by organizations such as the Jet Propulsion Laboratory (JPL) [6]. MBSE in combination with STECA-Priv constitutes a potentially powerful tool-based approach to the risk-aware design of privacy-sensitive systems.

The remainder of the paper is organized as follows. Section II provides background on STAMP, STECA, and MBSE. Section III discusses STECA and the modifications for privacy that result in STECA-Priv. Section IV uses the example of a smart TV to illustrate the combined use of STECA-Priv and MBSE. Section V situates STECA-Priv with respect to some related work while Section VI presents some concluding thoughts.

II. Background

A. STAMP

STAMP frames safety in terms of constraints rather than events [2]. Safety is achieved through the proper enforcement of complete and correct constraints on system behavior, rather than the prevention of certain events or chains of events. The more inclusive notion of behavioral constraints has the potential to identify problems arising out of issues such as

Approved for Public Release; Distribution Unlimited. Case Number 17-0621

unanticipated component interactions. These constraints are what controls enforce.

Controls are structured hierarchically with controls at each level enforcing constraints on processes in the level below it. This control structure exhibits multiple aspects. As described by Leveson [2], controls invariably involve adaptive feedback mechanisms, i.e., they are closed-loop controls. (Some privacy controls, though, lack feedback loops, i.e., they are open-loop controls.) Communication channels carry control commands to the relevant processes and information from the processes to the controllers. Accidents can result from four different types of control errors:

- Incorrect control action
- Missing control action
- Control action provided at the wrong time
- Incorrect duration of control action

For a control to properly constrain a process, it must maintain a model of that process. Control errors arise when the process model being used by a controller doesn't properly correspond to the process being controlled. (This can happen either because there is an error or gap in the model or because the model's state does not match the actual process state.)

B. STPA

STPA is a safety risk analysis methodology based on the concepts of STAMP. It was adapted for security (STPA-Sec) [3] prior to being adapted for privacy [1]. Its application in all its variants, however, requires a reasonably fleshed out system specification or description, such that the relevant control structure can be extracted and analyzed. This presents obvious problems when in the early, conceptual stages of the SELC. STECA [4] is a response to this problem and aims to infer a required control structure by extracting control concepts from an early-stage system description, specifically a ConOps that describes envisioned system behavior at a high level without specifying how that behavior will be achieved. STECA effectively attempts to deconstruct the system description to determine how applicable behavioral constraints could be enforced. Whereas STPA is fundamentally an analytical technique (answering the question "What is going on here?"), STECA is fundamentally an instrumental one (answering the question "What should be done here?").

C. MBSE

Because STECA aims to project a nominal functional control structure consistent with the system description and relevant constraints, it lends itself to MBSE as a means of describing and assessing that control structure. MBSE has been driven in part by a 2007 joint initiative of INCOSE and OMG [6]. This initiative (part of the larger INCOSE SE Vision 2020) defines MBSE as "the formalized application of modeling to support system requirements, design, analysis, verification and validation activities beginning in the conceptual design phase and continuing throughout development and later life cycle phases" and notes the development of mathematical foundations in 1993 [7]. MBSE constitutes an effort to shift systems engineering from a document-centric to a model-centric approach, a reaction to the effort required to develop and maintain traditional SELC documentation, including requirements and design specifications, as well as to the increasing difficulty of understanding complex socio-technical systems at even a conceptual level. In MBSE, the model serves as the central artifact and traditional SELC documentation is automatically generated from the model, more efficiently and effectively maintaining currency and consistency. Simulation via executable models enables MBSE to be leveraged across the SELC.

A variety of powerful tools are available that support MBSE using various modeling languages, including UML and SysML. For our purposes, SysML is preferable due to how it handles constraints. Constraints in UML (specified using Object Constraint Language) are annotations, i.e., they convey information to the engineer but are not operationally integrated within the model. SysML constraints, in contrast, are integrated into model execution so that as it executes, constraint violations can be observed and captured.

This capability is essential to extracting full benefit from using MBSE in conjunction with STECA. MBSE using SysML will reveal aspects of the projected control structure that could potentially result in constraint violations and thus present risks. Those aspects of the projected control structure can then be reconsidered and the risks represented by the constraint violations appropriately managed through mitigation, avoidance, transfer, or explicit acceptance. Any resulting changes to the control structure are captured by the model and their effects verified through its execution.

III. From STECA to STECA-Priv

As previously noted, STECA aims to project a system control structure from a description of system behavior and a safety risk model expressed in the form of system behavioral constraints. Developed by Fleming [4], STECA consists of six major (not strictly linear and potentially iterative) steps divided into ConOps analysis and safety-driven design:

1. Identify system hazards (ConOps analysis)
2. Derive system safety constraints (safety-driven design)
3. Identify control concepts (ConOps analysis)
4. Identify hazardous scenarios and causal factors (ConOps analysis)
5. Derive refined safety constraints (safety-driven design)
6. Refine, modify control structure (safety-driven design)

In the same way that STECA leverages the activities developed for STPA, in adapting STECA for privacy we can avail ourselves of the adaptations of STPA developed for STPA-Priv. Indeed, the first several steps are largely identical to STPA-Priv (and we refer the reader to [1] for detailed explanations of them):

1. Identify potential adverse privacy consequences to be considered, as denoted by a selected framework

2. Identify vulnerabilities that can lead to adverse privacy consequences in the context of the system

3. Specify system privacy constraints

As with STPA-Priv, we structure STECA-Priv to accommodate any of a variety of privacy frameworks (i.e., risk models), recognizing the pluralism of understandings of privacy in this regard.

In STPA-Priv, Step 3 combines the specification of privacy constraints with the specification of the system functional control structure. However, for STECA-Priv we must infer that structure, a less straightforward process. It is at this point, therefore, that STECA-Priv substantively departs from STPA-Priv:

4. Identify system privacy control concepts and infer privacy control models

5. Use MBSE to represent privacy control structure and constraints as an executable model to identify risks in the form of constraint violations and their causal factors, including malicious actions

OR

5. Manually analyze privacy control structure for

   a. Completeness

   b. Allocation of system privacy responsibilities

   c. Coordination and consistency

6. Revise privacy control structure and constraints

Note the absence of explicit reference to a ConOps. While one could be used for STECA-Priv if it exists, it is not strictly necessary. (Nor, arguably, is it strictly necessary for STECA). While substituting other types of high-level system description (nominal use cases or data flows, for example) may introduce additional difficulty, it does not fundamentally change the approach. By the same token, opting to represent the control structure and privacy constraints as an executable model doesn't fundamentally change the approach either. Rather, it enhances it by enabling the identification and management of the privacy risks represented by constraint violations, a more focused lens for viewing the conceptual system than the less precise characteristics in the alternative Step 5. Table I shows how the steps of STECA-Priv compare with the steps of STECA, assuming use of MBSE.

IV. Applying STECA-Priv

As before [1], we will use the example of a smart television to illustrate the application of the methodology. However, unlike the STPA-Priv example, which analyzed an existing smart TV implementation (as inferred from its privacy policy), STECA-Priv will be used to support the conceptual stage of engineering a smart TV. To keep the example small, we will focus on a distinct subset of the TV's operations: the collection and use of viewing data, managing consent for this collection and use, and managing the governing privacy policy on the device.

TABLE I. STECA-Priv Versus STECA

STECA-Priv	STECA
Identify potential adverse privacy consequences to be considered, as denoted by a selected framework	Identify system hazards
Identify vulnerabilities that can lead to adverse privacy consequences in the context of the system	
Specify system privacy constraints	Derive system safety constraints
Identify system privacy control concepts and infer privacy control models	Identify control concepts
Use MBSE to represent privacy control structure and constraints as an executable model to identify risks in the form of constraint violations and their causal factors, including malicious actions	Identify hazardous scenarios and causal factors
Revise privacy control structure and constraints	Derive refined safety constraints
	Refine, modify control structure

The smart TV will collect and store a record of the programs watched on the device. This viewing data will consist of entries that include the date and time (timestamp), the channel number selected, the program name, and the service provider (to enable matching the channel number to a specific network). This viewing data will be regularly transmitted to the TV manufacturer, who will combine it with demographic data (based on IP address) and maintain the combined data, including IP address, in a repository. Users must opt-in (provide explicit consent) to the collection and use of viewing data by the manufacturer, as governed by the privacy policy associated with the TV.

For the purposes of this example, the preceding paragraph will function as the system description that serves as input to STECA-Priv.

A. *Step 1: Identify potential adverse privacy consequences to be considered, as denoted by a selected framework*

As we did when illustrating the use of STPA-Priv previously, we will use Calo's subjective/objective privacy harms [8] as the framework for identifying adverse privacy consequences. Our choice of this framework is motivated by its relative simplicity, a useful characteristic for demonstration purposes. For a real-world project, a different or additional framework, including Fair Information Practice Principles (FIPPs), almost certainly would be used. A subjective privacy harm is the perception of unwanted surveillance. An objective privacy harm is the forced or unanticipated use of personal (i.e., specifically related to a person) information.

B. *Step 2: Identify vulnerabilities that can lead to adverse privacy consequences in the context of the system*

The use of explicit consent goes a long way toward avoiding the potential for subjective privacy harms. However, an objective privacy harm may result if that consent isn't accompanied by an accurate governing privacy policy that conveys the collection and use of viewing data and any relevant terms (e.g., regarding user access to the collected data). Similarly, an objective privacy harm may result if consent is not obtained or renewed following a change to that policy. This may be accompanied by a subjective privacy harm upon realization by a user of a material change to the policy absent consent. The relevant vulnerabilities can be expressed as:

1. The privacy policy associated with the smart TV is inaccurate as it pertains to viewing data collection and use.
2. Privacy policy and viewing data consent become unsynchronized.

C. *Step 3: Specify system privacy constraints*

These vulnerabilities can be reframed as corresponding system privacy constraints:

1. At any point in time, the privacy policy associated with the TV must be accurate.

This can be decomposed into two distinct constraints:

a. The privacy policy resident on the smart TV is the current privacy policy in effect for the smart TV.

b. The current privacy policy in effect for the smart TV correctly describes the applicable privacy practices.

2. Consent must correspond to the current privacy policy.

This also can be decomposed into two distinct constraints:

a. Consent must be specific to the current smart TV privacy policy.

b. Viewing data may be transmitted only if consent has been registered.

To this point, the process has been the same as if we were applying STPA-Priv to the described system. But, of course, this is not a fully described or specified system, but a (circumscribed) high-level concept which must be fleshed out. It is at this point that STECA-Priv diverges from STPA-Priv.

D. *Step 4: Identify system privacy control concepts and infer privacy control models*

In the absence of a more or less complete system specification, we must infer *a* functional privacy control structure rather than extract *the* control structure. The initial step in performing this inference is identifying the system privacy control concepts from the limited description that we have. To accomplish this, we assign control loop roles to applicable elements in the system description. Figure 1 shows the general form of a STAMP control loop. A STAMP control loop includes four principal roles: controller, actuator, controlled process, and sensor. The controller issues control actions, which are executed by some kind of (not necessarily physical) actuator, which acts on the controlled process, the relevant behavior of which is conveyed back to the controller by the sensor.

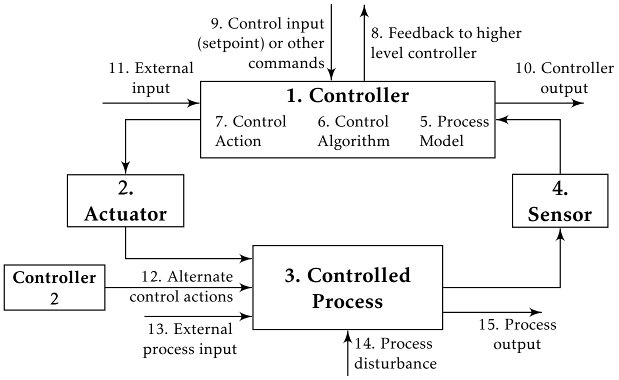


Fig. 1. STAMP Control Loop [4]

Once sets of these elements have been identified, the privacy control model corresponding to each set can be developed. (It is not expected that all potential model elements will be specified; not all elements are mandatory in a control loop and limited information may preclude identifying or projecting others.) Table II provides questions to guide this process. This yields the control models in Tables III and IV, addressing constraints 1.a and 1.b respectively, and Tables V and VI, addressing constraints 2.a and 2.b respectively. To save space, these tables only contain those rows with populated descriptions.

Having inferred these control models, we can now use MBSE to represent the functional privacy control structure of these aspects of the system together with the applicable privacy constraints. To keep the example manageable, we will focus on the control loops implied by Tables V and VI in which the smart TV is the controller. These control models address constraint 2.

E. *Step 5: Use MBSE to represent control structure and constraints as an executable model to identify risks in the form of constraint violations and their causal factors, including malicious actions*

We begin this process by creating the overall structure of the model. Figure 2 shows this structure using SysML block definition diagrams. The principal blocks represent the smart TV and the smart TV manufacturer. (Although denoted separately, the TV viewing dataset is part of the smart TV manufacturer.) We have also included the smart TV user and the demographics provider for context. (As a practical matter, the modeler will play the role of the user.) The specified multiplicities are nominal and do not affect the example. Communications between the user (actually the modeler) and the TV and the TV and manufacturer are implemented by SysML signals and events via ports that are denoted on internal block diagrams representing the TV and manufacturer blocks.

The remaining block is a constraint block, Consent-Policy Sync. Constraint blocks define conditions that should hold at all times. As long as the constraint expression evaluates to true, the constraint is satisfied. If the expression evaluates to false,

TABLE II. STAMP CONTROL MODEL ELICITATION [4]

	Control Role	Description
1.	Controller	Which controller is being described in the text?
2.	Actuator	What mechanism(s) does the control have in order to affect the process?
3.	Controlled Process	What process does the controller have control over?
4.	Sensor	What type of feedback does the controller receive about the process it controls?
5.	Process Model	What states and variables does the controller know about the process it controls?
6.	Control Algorithm	Does the controller use an algorithm or procedure to generate action?
7.	Control Actions	What types of action can the controller generate?
8.	Controller Status	Does the controller provide feedback to higher level controllers?
9.	Control Input	Does the controller receive set points or other types of commands?
10.	Controller Output	Does the controller have output other than through the actuator? This often includes transmission of information to other controllers.
11.	External Controller Input	Does the controller receive external input, either in terms of other system information or other controller action(s), or other (e.g., a power source)?
12.	Alternate Control Actions	Does the process receive action from controllers other than in items 1 and 2?
13.	External Process Input	Does the process require external input to function? Examples include pressure, power, and heat.
14.	Process Disturbance	What environmental factors does the process interact with?
15.	Process Output	Does the system require that the process output something to other components? (e.g., power, pressure)

TABLE III. PRIVACY CONSTRAINT 1.A CONTROL MODEL

	Control Role	Description
1.	Controller	Smart TV manufacturer
2.	Actuator	Server
3.	Controlled Process	Update smart TV privacy policy
6.	Control Algorithm	Transmit smart TV privacy policy when current policy changes
7.	Control Actions	Transmit current smart TV privacy policy
15.	Process Output	Smart TV privacy policy

TABLE IV. PRIVACY CONSTRAINT 1.B CONTROL MODEL

	Control Role	Description
1.	Controller	Smart TV manufacturer
2.	Actuator	Data governance compliance process
3.	Controlled Process	Smart TV privacy policy implementation
6.	Control Algorithm	Periodic and triggered review
7.	Control Actions	Revise smart TV privacy policy to conform to practices
15.	Process Output	Smart TV privacy policy

TABLE V. PRIVACY CONSTRAINT 2.A CONTROL MODEL

	Control Role	Description
1.	Controller	Smart TV
3.	Controlled Process	Privacy policy update
5.	Process Model	Policy updated indicator (policyUpdated)
6.	Control Algorithm	Determine consent if privacy policy is updated by manufacturer
7.	Control Actions	Receive policy update; process policy update
9.	Control Input	Smart TV privacy policy
12.	Alternate Control Actions	Grant/deny consent
15.	Process Output	Displayed privacy policy and consent option

TABLE VI. PRIVACY CONSTRAINT 2.B CONTROL MODEL

	Control Role	Description
1.	Controller	Smart TV
3.	Controlled Process	Viewing data transmission
5.	Process Model	Consent indicator (consent)
6.	Control Algorithm	Periodically send viewing data if consent has been granted
7.	Control Actions	Send viewing data
11.	External Controller Input	Consent granted/denied
15.	Process Output	Viewing data | consent granted; no output | consent denied

the constraint has been violated. While primarily intended to apply to models involving substantial mathematical calculation, we are using this facility to define a logical constraint using Boolean variables, which are described below.

The control models described by Tables V and VI are represented by the state machine associated with the smart TV block and shown in Figure 3. As a practical matter, control models and their representations may require interative refinement. Within the control representation, control actions take the form of SysML activities. Thus, the activities called in the state machine diagram correspoind to the similarly named control actions described in Tables V and VI.

The process models corresponding to constraint 2 are represented using Boolean variables whose values indicate whether a privacy policy update has been received from the smart TV manufacturer (2.a) and whether consent for the collection and use of viewing data has been granted (2.b). To avoid viewing interruption, policy updates are presented to the user when the TV is turned on, requiring two distinct control actions for receiving and processing policy updates. Note that because a policy update is registered as the default value (**policyUpdated == true**), this will happen “out of the box.”

If **policyUpdated == true**, the TV channel is switched to channel 0 (the TV’s user messaging channel) to display the updated policy and the policy update indicator is cleared (**policyUpdated = false**). The user must then grant or deny consent, which sets the consent indicator accordingly. The TV then enters its Operating state. (If **policyUpdated == false**, the TV moves immediately from the On to the Operating state.) If consent has been granted (**consent == true**), the TV

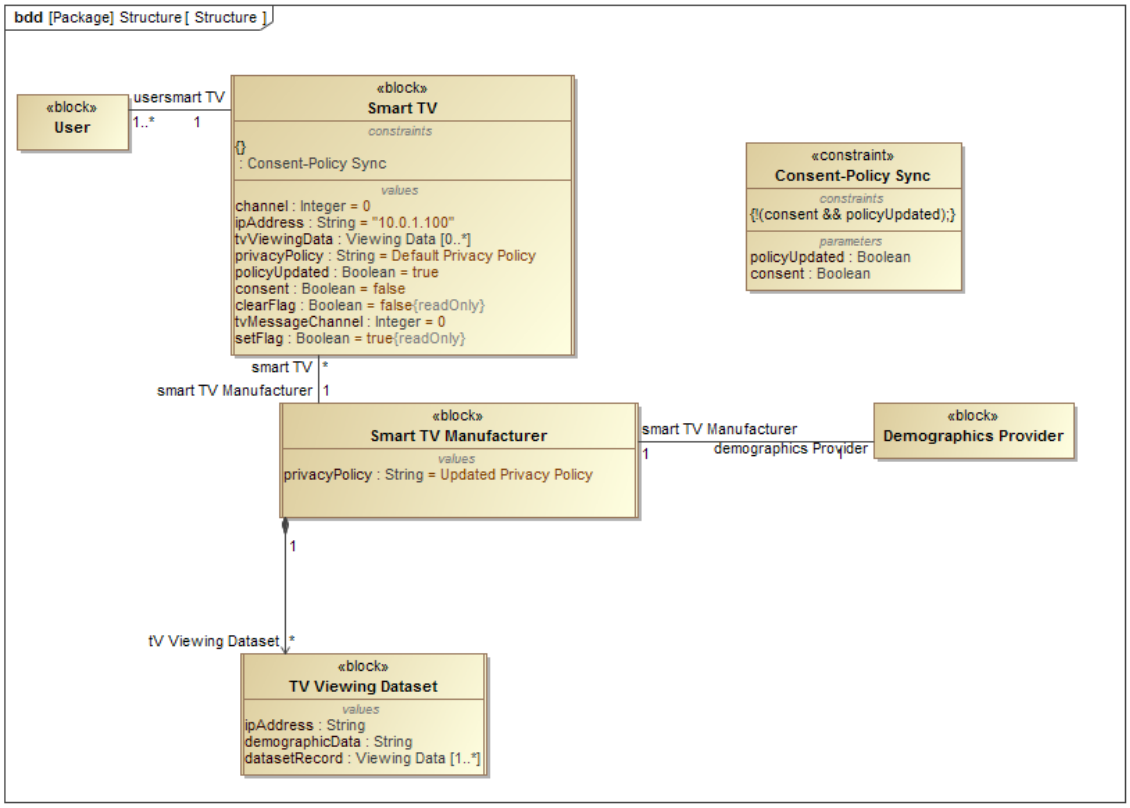


Fig. 2. Structure of Smart TV Model

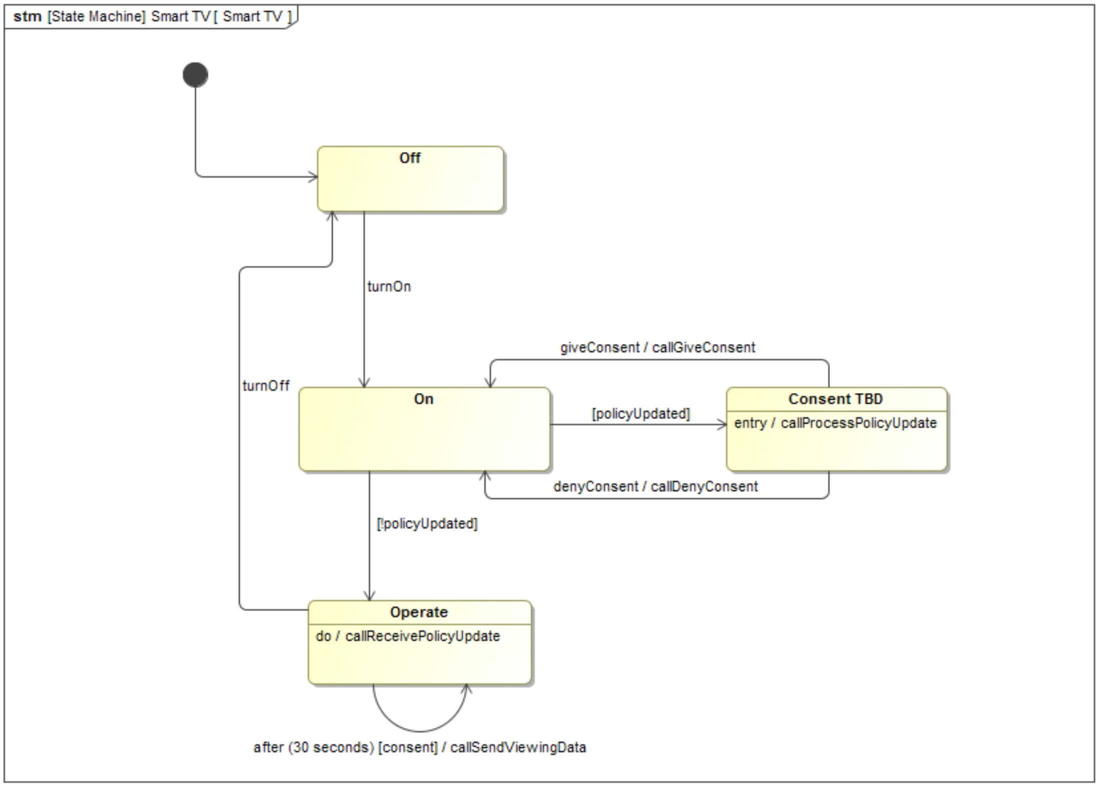

stm [State Machine] Smart TV [Smart TV]
Off
turnOn
turnOff
giveConsent / callGiveConsent
On
[policyUpdated]
Consent TBD
entry / callProcessPolicyUpdate
denyConsent / callDenyConsent
[!policyUpdated]
Operate
do / callReceivePolicyUpdate
after (30 seconds) [consent] / callSendViewingData

Fig. 3. Smart TV State Machine Diagram

periodically transmits viewing data to the manufacturer, where it is processed and stored. Thus, policyUpdated acts as a transition guard in the On state while consent acts as a transition guard in the Operating state. While in the Operating state, the TV monitors for privacy policy updates and, if one is received, replaces the previous policy with the new one and sets the policy update indicator (**policyUpdated = true**).

The constraint reflects the fact that at no time should there be an updated policy indicator (**policyUpdated == true**) together with affirmative consent (**consent == true**). If this occurs, it implies that an updated privacy policy has not yet been presented to the user for them to consent to, but that consent has nevertheless been granted. This would constitute a violation of constraint 2.a. Note that constraint 2.b is enforced by the guard condition on the Operating state transition that results in the transmission of viewing data.

As the left window of Figure 4 shows, this constraint was violated ("failed") when the model was executed. Execution involved turning the TV on, granting consent, then updating the privacy policy. The right window shows the implicated variables shaded red. (While the constraint held, they were shaded green.) Upon examination, the problem becomes apparent. Because an updated privacy policy is not immediately presented to the user and the consent indicator is not cleared when the policy update indicator is set, viewing data continues to be transmitted. In other words, if the user had previously consented to the collection and use of viewing data by the manufacturer, that consent would continue under a new policy until such time as the TV was turned off and then on again.

F. Step 6: Revise privacy control structure and constraints

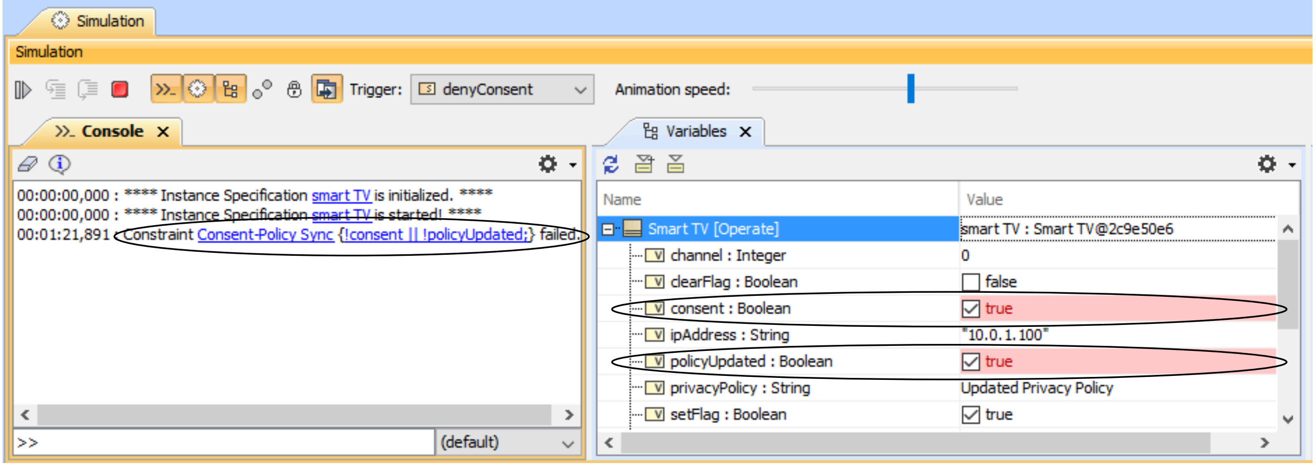
From a STAMP perspective, this represents a control action provided at the wrong time. The most straightforward way of addressing this (thereby mitigating the privacy risk related to the constraint violation), is to clear the consent indicator (**consent = false**) at the same time as the policy update indicator is set (**policyUpdated = true**). This prevents viewing data from being transmitted under the new policy until the user has had the opportunity to grant or deny consent. This is verified by making the necessary changes executing the modified model, and observing that the constraint failure no longer occurs (and the relevant variables in the right window of Figure 4 are shaded green rather than red).

V. Related Work

STECA-Priv, like STPA-Priv, bears some relationship to goal-oriented modeling [9], since it implicitly deals with goals in the form of constraints and obstacles in the form of problematic control actions. Further, there has been discussion of leveraging MBSE in support of goal-oriented modeling [10], though it is unclear what, if anything, has resulted from this. There has also been work using modeling to support Privacy by Design based on a specific privacy framework [11], including automating OCL constraint checking [12].

Also like STPA-Priv, but unlike goal-oriented modeling, STECA-Priv takes an approach explicitly grounded in systems theory. This manifests itself not only in a typology of problematic control actions, but also in the use of role-based control models. Together with the use of a chosen privacy framework, these contribute to a more focused and practically-structured methodology. As such, it is a methodology that strikes a balance between rigidity and open-endedness.

This balance is also reflected in the use of MBSE in support of STECA-Priv. By definition, MBSE is a general, open-ended process, as is reflected by some of its specific methods [13]. However, because the control models derived from the application of STECA-Priv drive the MBSE process, it is circumscribed in a way not characteristic of MBSE methods more generally. However, owing in part to the limits of reasonable inference, there still will be



Trigger: denyConsent

```
00:00:00,000 : **** Instance Specification smart TV is initialized. ****
00:00:00,000 : **** Instance Specification smart TV is started! ****
00:01:21,891 : Constraint Consent-Policy Sync {!consent || !policyUpdated;} failed.
```

Name	Value
Smart TV [Operate]	smart TV : Smart TV@2c9e50e6
channel : Integer	0
clearFlag : Boolean	□ false
consent : Boolean	☑ true
ipAddress : String	"10.0.1.100"
policyUpdated : Boolean	☑ true
privacyPolicy : String	Updated Privacy Policy
setFlag : Boolean	☑ true

Fig. 4. Constraint Violation During Model Execution

significant degrees of freedom available to the engineer. Striking this balance, one way or another, is historically necessary for effective engineering praxis [14] within a given engineering discipline. Insufficient focus leaves practitioners struggling for traction, while an overly prescriptive approach suffers from limited efficacy.

VI. Conclusion

STECA-Priv is an instrumental method for performing privacy risk management on complex socio-technical systems at early life cycle stages. It does not require a relatively complete system description, instead inferring provisional privacy control structure in a systematic fashion for the purpose of system specification. Note, though, that the use of STECA-Priv in early life cycle stages does not obviate the need for downstream risk analysis, such as STPA-Priv, once the system is more fully fleshed out; earlier assumptions and inferences may no longer be valid and/or other privacy risks may have been introduced. We are currently applying STECA-Priv to a real-world identity management project, which should provide insights into its utility and practicality. As it is a new method, modifications based on that experience may be needed, though we are confident that the fundamental soundness of the approach will be validated.

Clearly, we also believe in the potential value of combining STECA-Priv with MBSE. The ability to model a system's inferred privacy control structure and to execute that model with integrated constraints adds further rigor to the application of STECA-Priv. It supports identification and correction of privacy control problems early in the SELC. While STECA is premised on the availability of a system ConOps, this is arguably an arbitrary target. Whether sufficient descriptive information exists to justify the effort required to apply STECA-Priv will be a case-by-case judgment call.

So too will be the use of MBSE in conjunction with STECA-Priv. Our experience with a representative tool has been that the learning curve is steep. Even once that learning curve has been climbed, developing an executable SysML model of a complex system will require a substantial resource commitment, more so if the investment is to be maintained over time by capturing all future system changes in the model. However, we believe that the potential value of such models may be significant, enabling the detection of privacy risks, including emergent system properties, initially and as the system undergoes changes. While our illustrative example is relatively straightforward, real-world complex systems are decidedly less so. We anticipate MBSE typically would reveal unexpected privacy control problems. This would be especially valuable (and worth the resource expenditure) for systems that are intrinsically high risk due to their data, technology, and/or usage contexts.

Further work is needed to develop criteria for when an explicit constraint in the form of a constraint block is or isn't needed. In the case of the former, it would also be desirable to develop guidance on how to formulate such constraints based on the control models as represented in SysML. That representation involves a translation from the relative abstractions of the STECA-Priv control models to the more specific constructs of SysML. Such guidance actually might be considered a subset of broader guidance regarding how to perform that translation. If such guidance can be developed, the application of STECA-Priv in conjunction with MBSE would become less of an art and more of a discipline.

Acknowledgment

Thanks to MITRE colleague Julie Snyder and to an anonymous reviewer for comments that signficantly improved this paper.

References

[1] S. Shapiro, "Privacy Risk Analysis Based on System Control Structures: Adapting System-Theoretic Process Analysis for Privacy Engineering," 2016 IEEE Security and Privacy Workshops (SPW), San Jose, CA, pp. 17-24, 2016.

[2] N. G. Leveson, Engineering a Safer World: Systems Thinking Applied to Safety. Cambridge, MA: MIT Press, 2011.

[3] W. Young and N. G. Leveson, "An integrated approach to safety and security based on systems theory," Commun. ACM, vol. 57, pp. 31-35, February 2014.

[4] C. H. Fleming, "Safety-driven Early Concept Analysis and Development" (PhD diss., Massachusetts Institute of Technology, 2015). Available at http://sunnyday.mit.edu/Fleming-dissertation-final.pdf.

[5] Model-Based Systems Engineering (MBSE) Wiki, http://www.omgwiki.org/MBSE/doku.php, accessed February 8, 2017.

[6] D. Nichols and C. Lin, "Integrated Model-Centric Engineering: The Application of MBSE at JPL Through the Life Cycle," INCOSE International MBSE Workshop, Los Angeles, CA, January 26, 2014. Available at http://www.omgwiki.org/MBSE/lib/exe/fetch.php?media=mbse:06-iw14-mbse_workshop-application_of_mbse_at_jpl_through_the_lifecycle-nichols-lin-final.pdf.

[7] A. W. Wymore, Model-Based Systems Engineering: An Introduction to the Mathematical Theory of Discrete Systems and the Tricotyledon Theory of System Design, Boca Raton, FL: CRC Press, 1993.

[8] M. R. Calo, "The boundaries of privacy harm," Indiana Law Journal, vol. 86, pp. 1131-1162, 2011.

[9] A. van Lamsweerde, Requirements Engineering: From System Goals to UML Models to Software Specifications. Chichester, UK: Wiley, 2009.

[10] J. C. Nwokeji, T. Clark and B. S. Barn, "Towards a comprehensive Meta-Model for KAOS," 2013 3rd International Workshop on Model-Driven Requirements Engineering (MoDRE), Rio de Janeiro, 2013, pp. 30-39.

[11] F. Jaime, A. Maña, Z. Ma, C. Wagner, D. Hovie and M. Bossuet, "Building a privacy accountable surveillance system," 2015 3rd International Conference on Model-Driven Engineering and Software Development (MODELSWARD), Angers, 2015, pp. 646-654.

[12] A. Kung, C. Jouvray and F. Coudert, "SALT frameworks to tackle surveillance and privacy concerns," 2015 3rd International Conference on Model-Driven Engineering and Software Development (MODELSWARD), Angers, 2015, pp. 665-673.

[13] T. Weilkiens, A. Scheithauer, M. Di Maio and N. Klusmann, "Evaluating and comparing MBSE methodologies for practitioners," 2016 IEEE International Symposium on Systems Engineering (ISSE), Edinburgh, 2016, pp. 1-8.

[14] S. Shapiro, "Degrees of Freedom: The Interaction of Standards of Practice and Engineering Judgment," Science, Technology, & Human Values, vol. 22, no. 3, 1997, pp. 286–316.