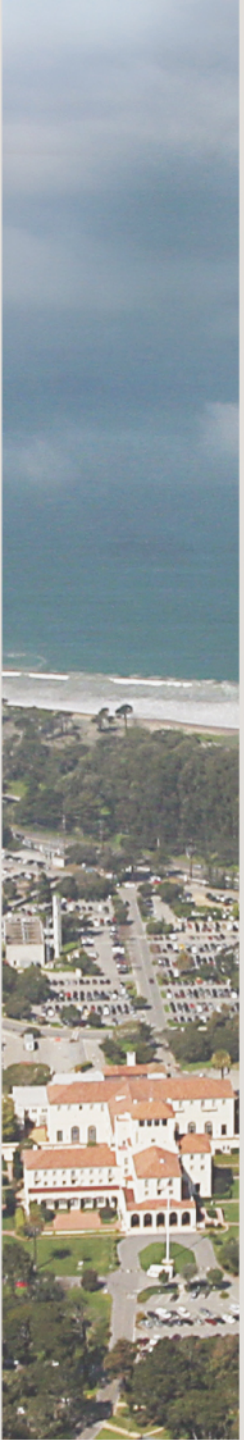


NAVAL
POSTGRADUATE
SCHOOL

Practical Modeling Concepts for Engineering Emergence in Systems of Systems

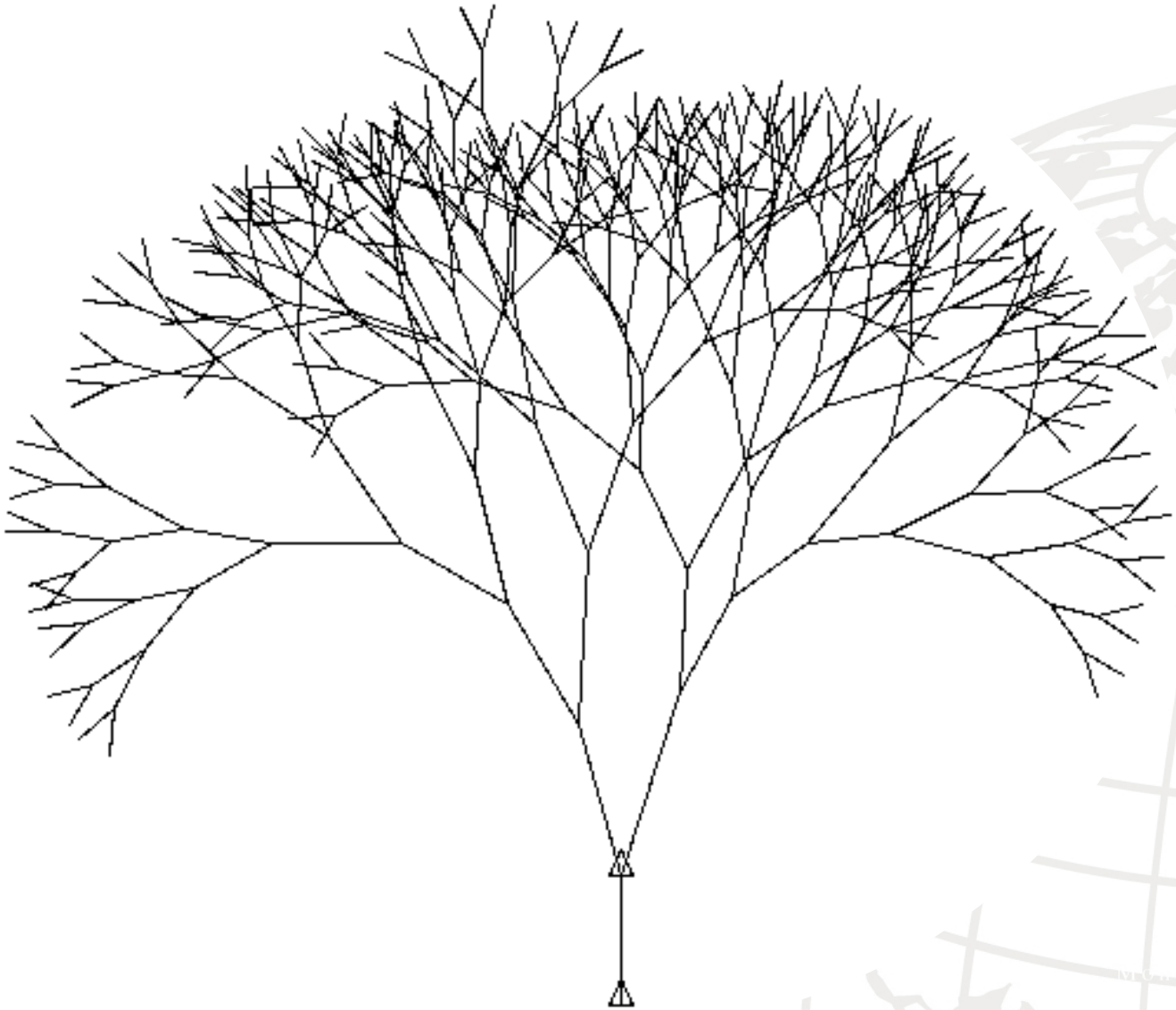
An Overview for the SoSECIE Webinar

Kristin Giammarco, Ph.D.
NPS Department of Systems Engineering
20 March 2018

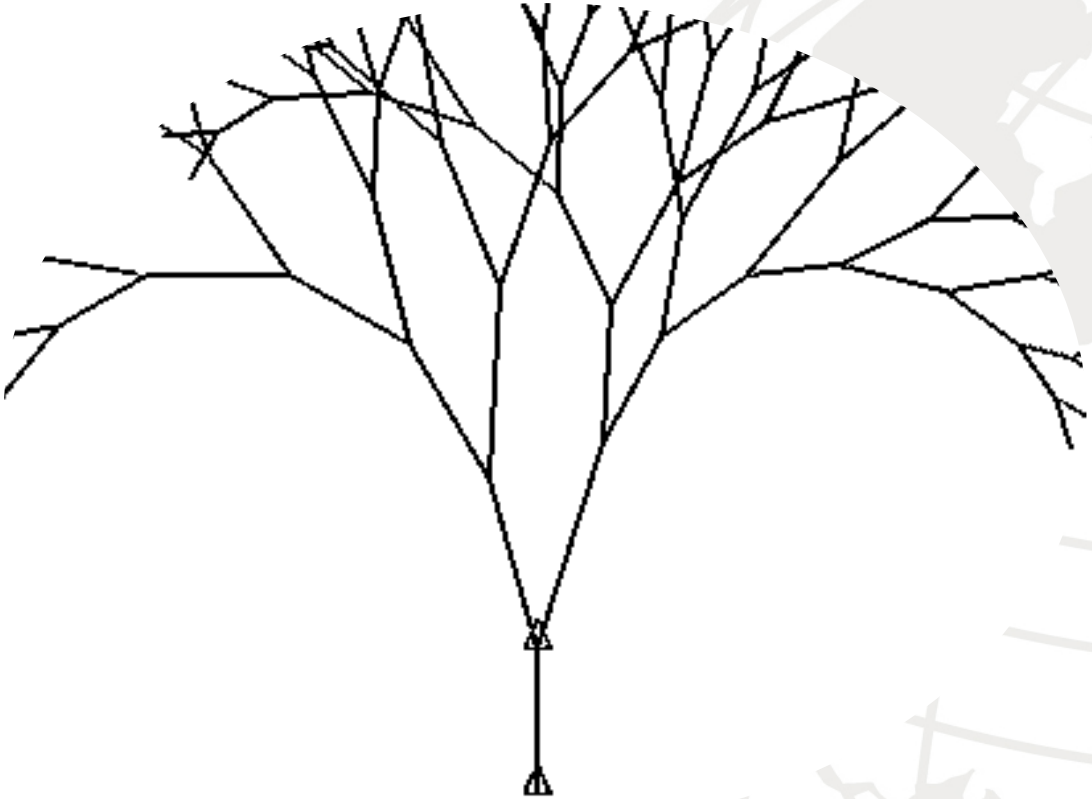
Monterey, California
WWW.NPS.EDU

Why Engineer Emergence?

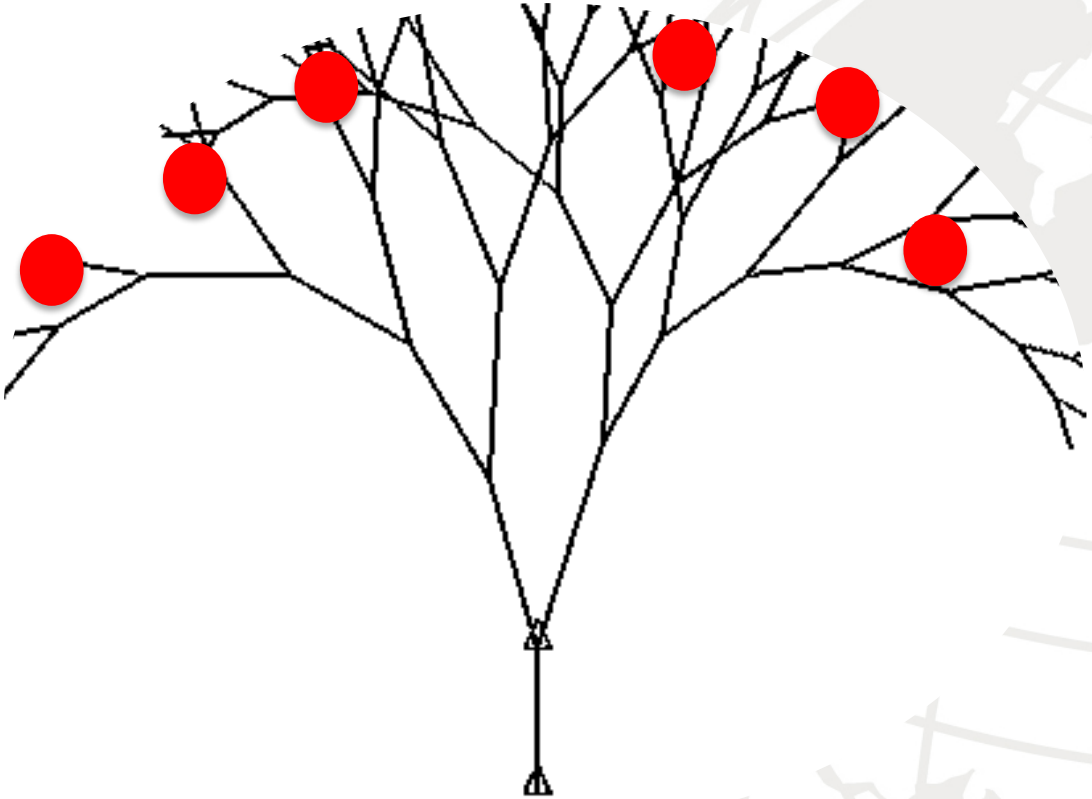
- Stakeholders desire their designed systems to exhibit “positive” emergent behaviors, and to suppress or exclude “negative” emergent behaviors
- How do we know what behaviors need to be suppressed or excluded from the design, before they actually emerge?



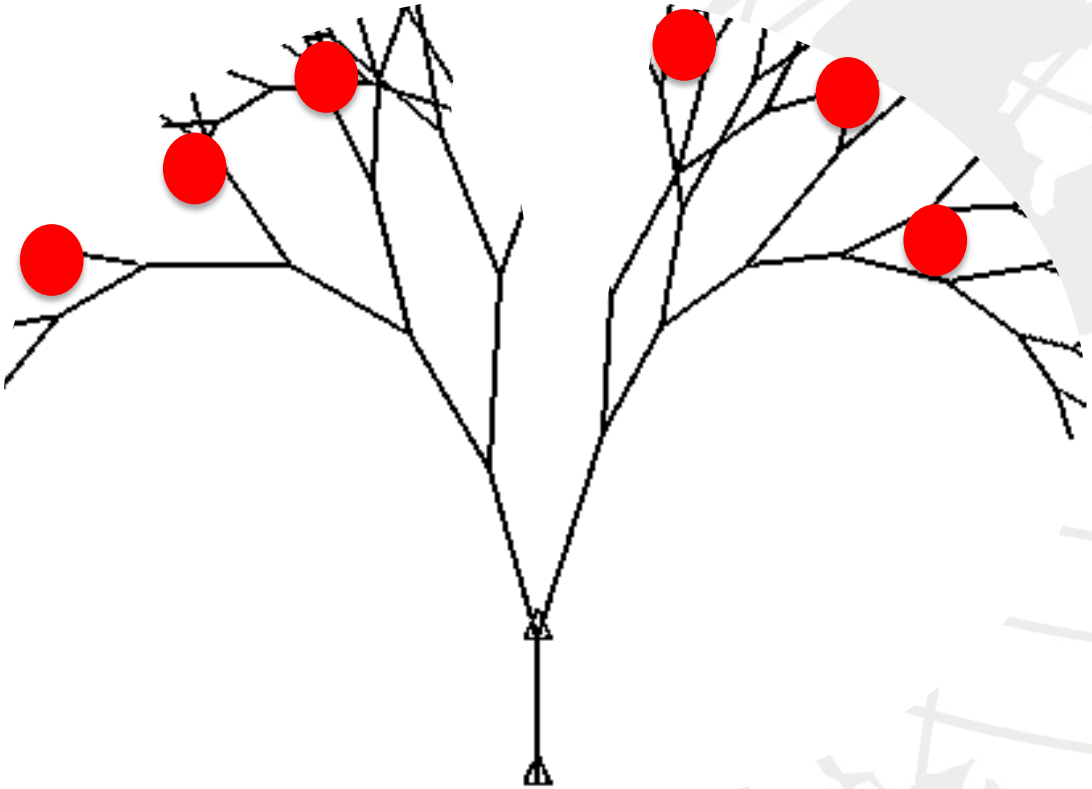
https://www.cs.unm.edu/~williams/cs257/graphics.html



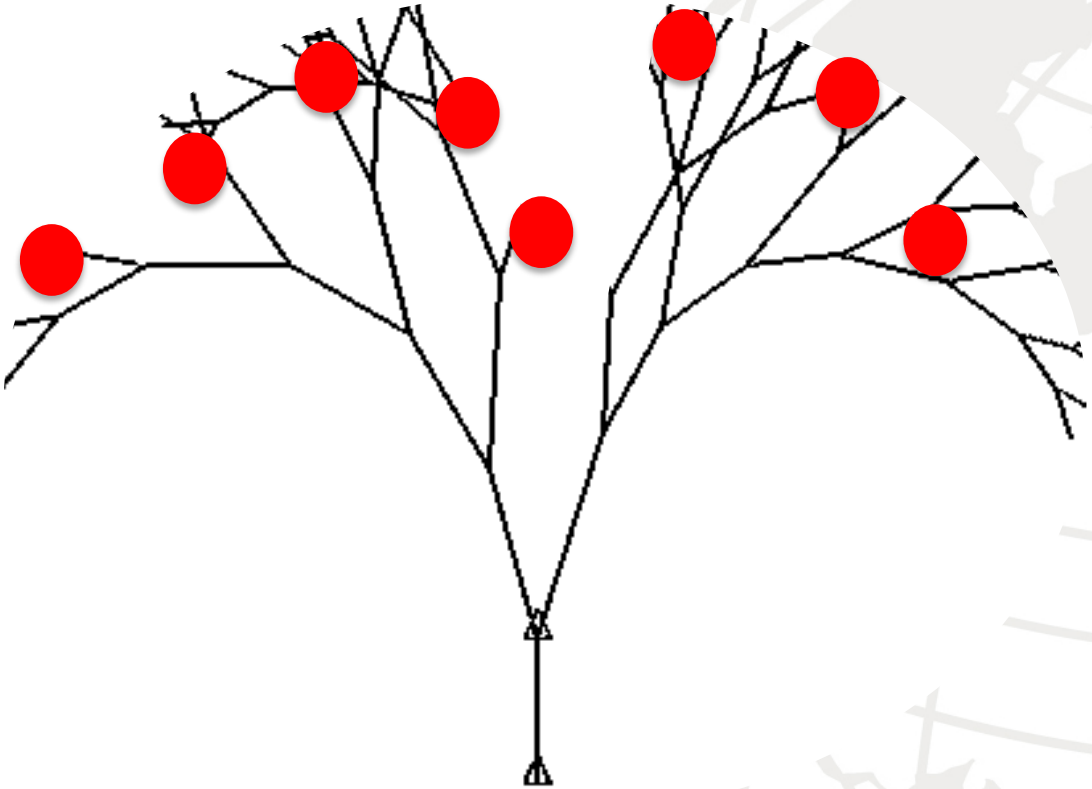
https://www.cs.unm.edu/~williams/cs257/graphics.html



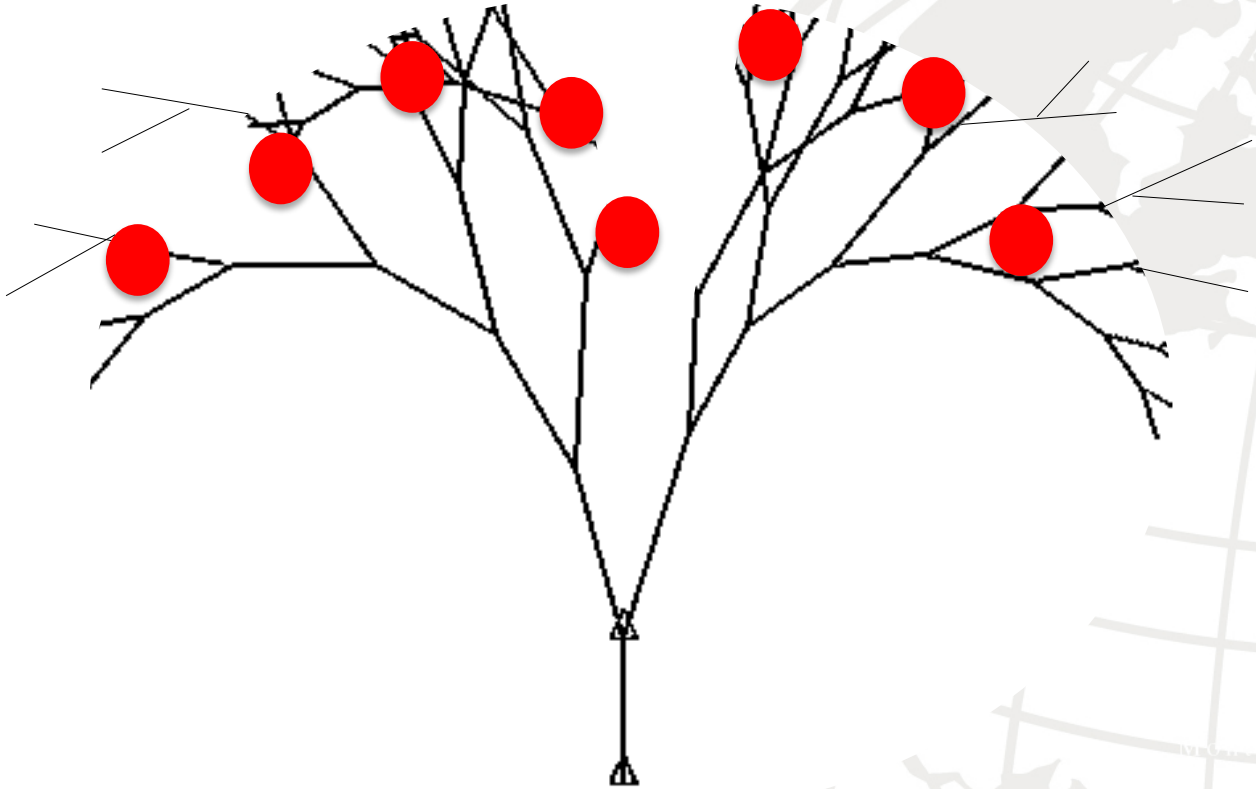
https://www.cs.unm.edu/~williams/cs257/graphics.html



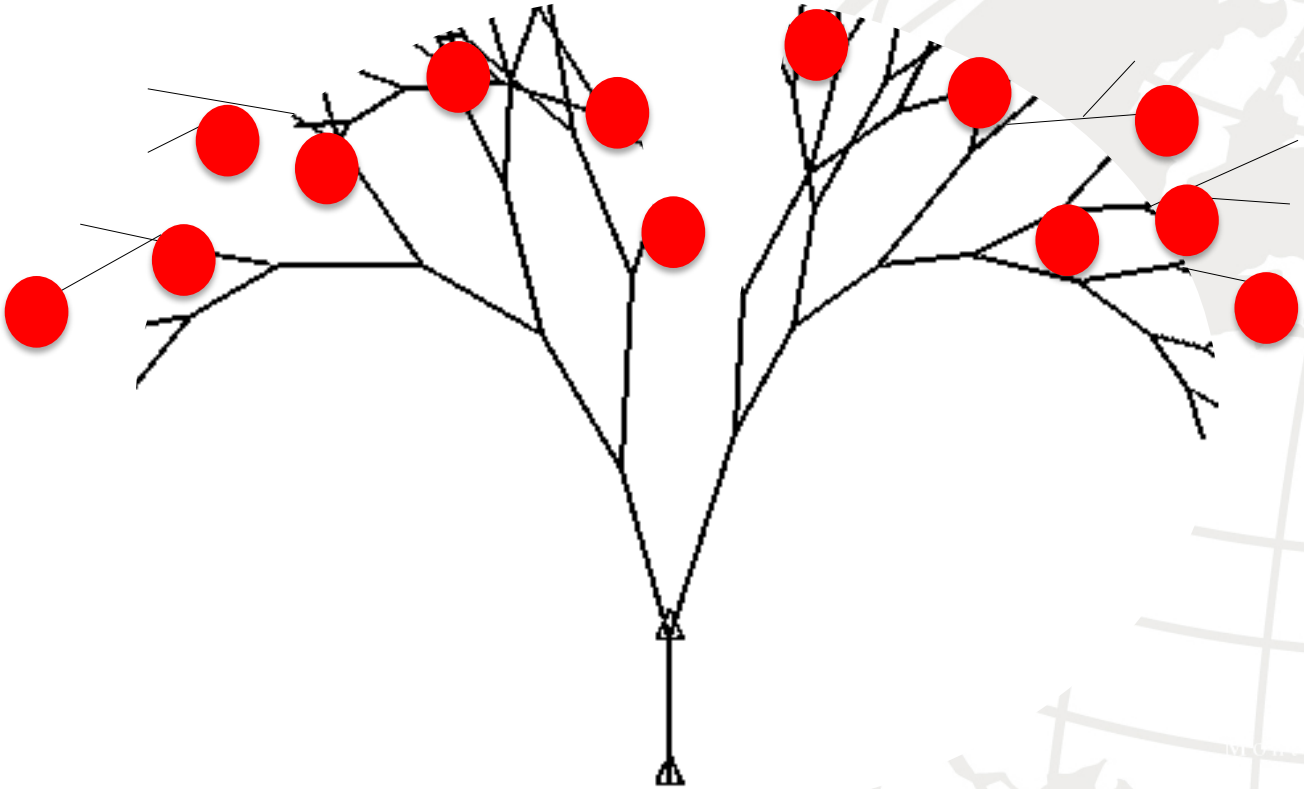
https://www.cs.unm.edu/~williams/cs257/graphics.html



https://www.cs.unm.edu/~williams/cs257/graphics.html

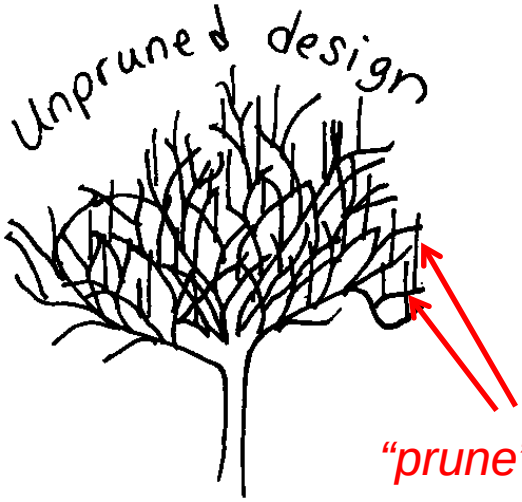


https://www.cs.unm.edu/~williams/cs257/graphics.html



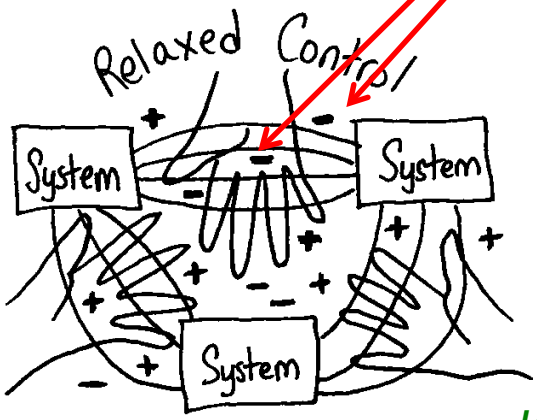
https://www.cs.unm.edu/~williams/cs257/graphics.html

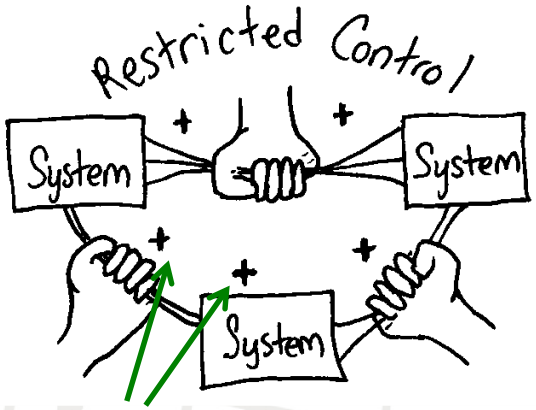
How do we “steer” behaviors in our complex systems?



"prune" objectionable behaviors

is to… as… is to…





leave behind only the desired behaviors

Positive emergence

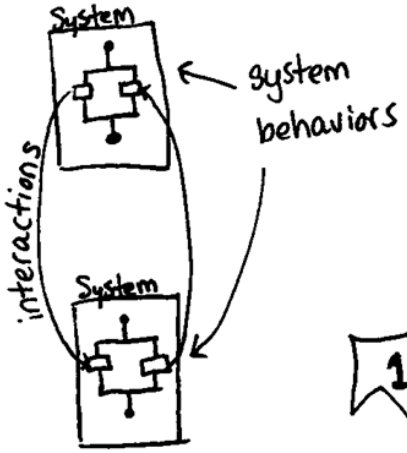
is what remains after thoroughly exposing and removing

Negative emergence.

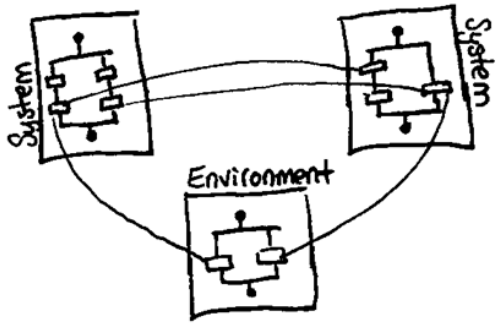
How do we do that?

Employ these modeling concepts

1. Separate system behaviors and interactions



2. Model system behaviors and environment behaviors



3. Formalize models for automatic execution

Run

MBSE

4. Properly allocate tasks to a

Human

error prone

lived experience
imagination
creativity

(brain)

- inspection
- evaluation
- pattern detection

or **Machine**

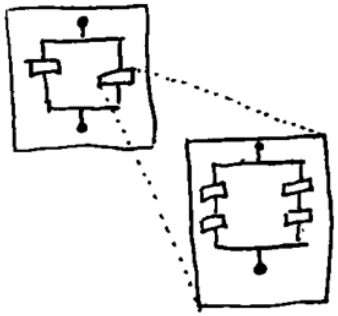
can't do what it's not programmed to do

- computation
- automation
- generation
- view projection

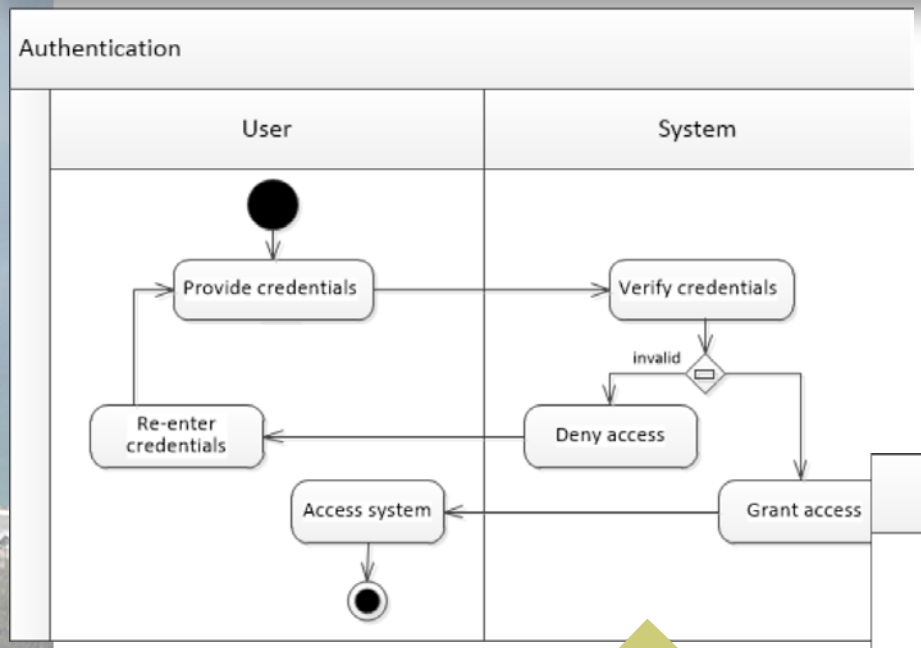
Searches on large data sets

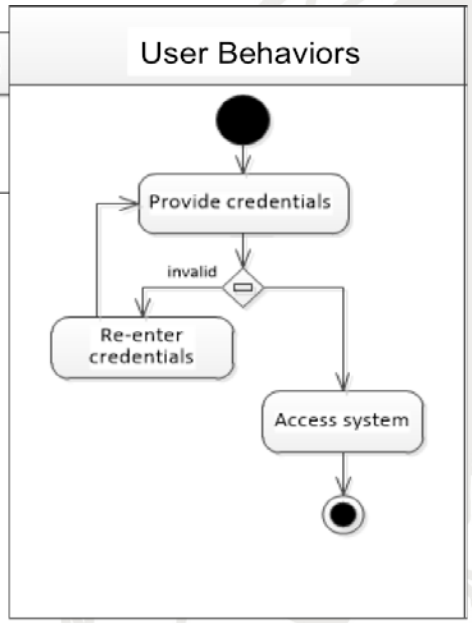
Next: Automate more of these functions

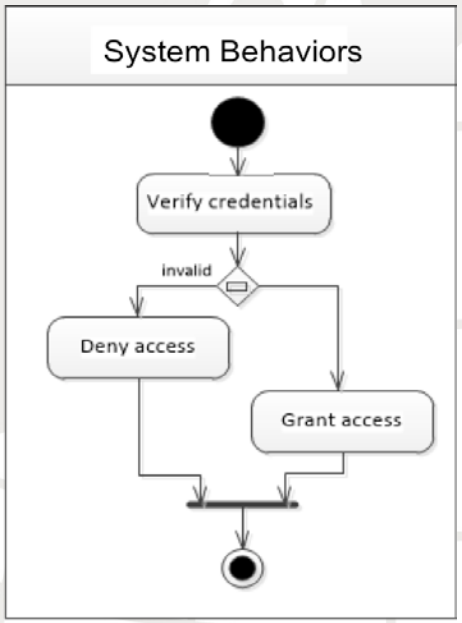
5. Use abstraction and refinement to manage large models.



1. Separate behaviors and interactions



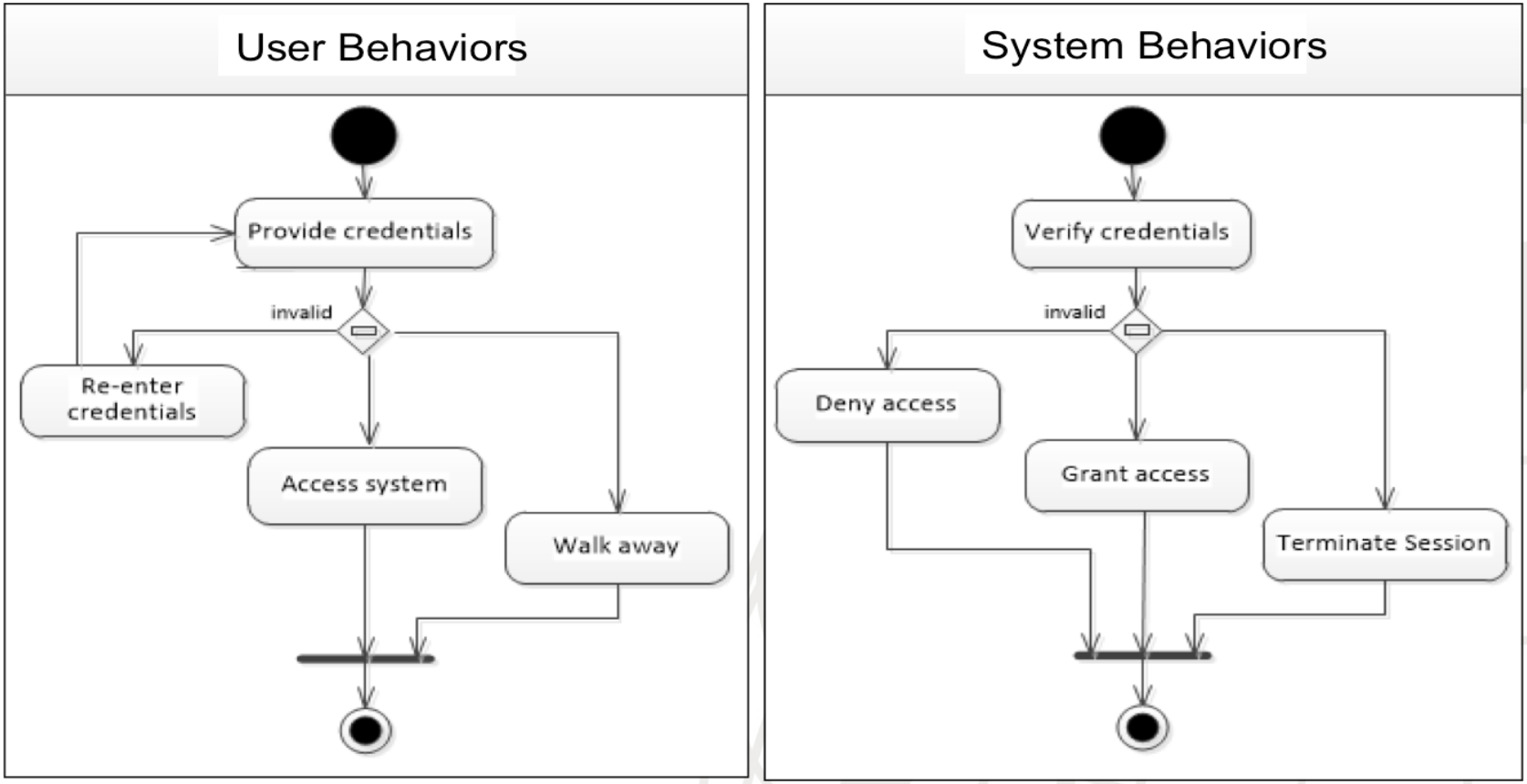




Interaction Constraints

- "Provide credentials" from the User precedes "Verify credentials" from the System
- "Deny access" from the System precedes "Re-enter credentials" from the User
- "Grant access" from the System precedes "Access system" from the User

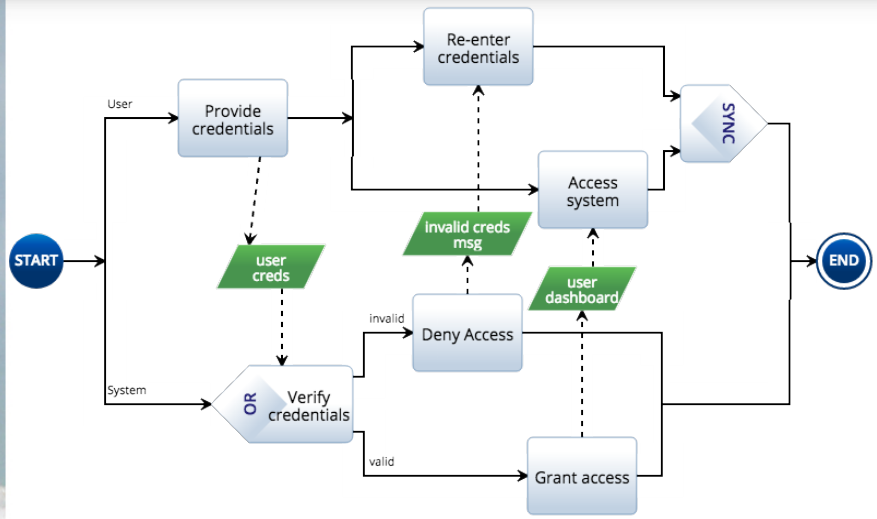
2. Model system behaviors and environment behaviors



Interaction Constraints

- "Provide credentials" from the User precedes "Verify credentials" from the System
- "Deny access" from the System precedes "Re-enter credentials" from the User
- "Grant access" from the System precedes "Access system" from the User
- "Walk away" from the User precedes "Terminate session" from the System

3. Formalize models for automatic execution




```
/*************************************************************************

A Model for Simple Authentication

  created by K.Giammarco on 05/16/2017
  modified by K.Giammarco on 08/07/2017 for capitalization of state events
  modified by K.Giammarco on 08/07/2017 for ENSURE constraints

*************************************************************************/

SCHEMA Authentication

/*-----------------------
  USER BEHAVIORS
-----------------------*/

ROOT User:  Provide_credentials
              (* CREDS_INVALID Reenter_credentials *)
            [ CREDS_VALID Access_system ];

/*-----------------------
  SYSTEM BEHAVIORS
-----------------------*/

ROOT System:    Verify_credentials
                (+ ( CREDS_INVALID Deny_access |
                     CREDS_VALID Grant_access ) +)
                [ Lock_account ]
                ;

/*-----------------------
  INTERACTION CONSTRAINTS
-----------------------*/

User, System SHARE ALL CREDS_VALID, CREDS_INVALID;

COORDINATE  $a: Provide_credentials FROM User,
            $b: Verify_credentials  FROM System
        DO ADD $a PRECEDES $b; OD;

COORDINATE  $a: Deny_access             FROM System,
            $b: Reenter_credentials     FROM User
        DO ADD $a PRECEDES $b; OD;

COORDINATE  $a: Grant_access        FROM System,
            $b: Access_system       FROM User
        DO ADD $a PRECEDES $b; OD;

ENSURE #CREDS_INVALID <= 3;

ENSURE #Deny_access >= 3 <-> #Lock_account == 1;

ENSURE #Grant_access >= 1 -> #Lock_account == 0;
```

4. Properly allocate each task to a human or to a machine

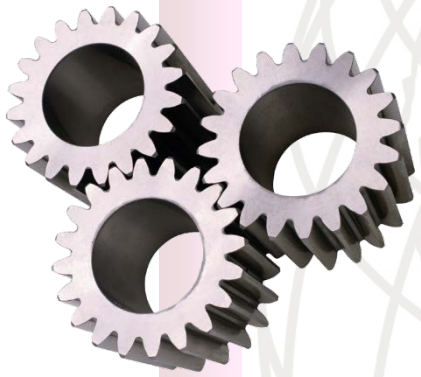
1. **Human** specifies system behaviors and interactions

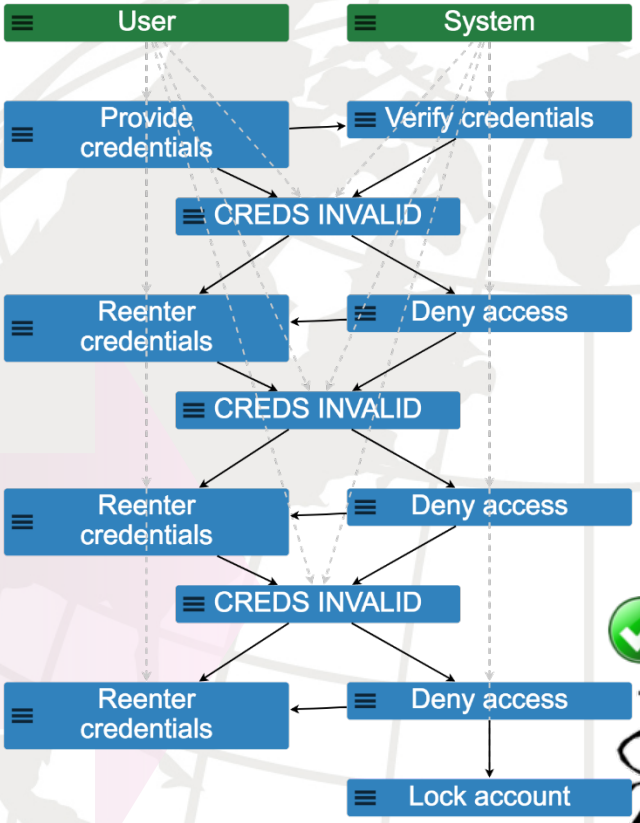
2. **Machine** generates SoS scenarios from the specification

3. **Human** conducts V&V on the automatically generated scenarios

```
SCHEMA Authentication

/*------------------------
  USER BEHAVIORS
------------------------*/

ROOT User:  Provide_credentials
            (* CREDS_INVALID Reenter_credentials *)
            [ CREDS_VALID Access_system ];

/*------------------------
  SYSTEM BEHAVIORS
------------------------*/

ROOT System:    Verify_credentials
                (+ ( CREDS_INVALID Deny_access |
                     CREDS_VALID Grant_access ) +)
                [ Lock_account ]
                ;

/*------------------------
  INTERACTION CONSTRAINTS
------------------------*/

User, System SHARE ALL CREDS_VALID, CREDS_INVALID;

COORDINATE  $a: Provide_credentials FROM User,
            $b: Verify_credentials  FROM System
        DO ADD $a PRECEDES $b; OD;

COORDINATE  $a: Deny_access             FROM System,
            $b: Reenter_credentials     FROM User
        DO ADD $a PRECEDES $b; OD;

COORDINATE  $a: Grant_access        FROM System,
            $b: Access_system       FROM User
        DO ADD $a PRECEDES $b; OD;

ENSURE #CREDS_INVALID <= 3;

ENSURE #Deny_access >= 3 <-> #Lock_account == 1;

ENSURE #Grant_access >= 1 -> #Lock_account == 0;
```

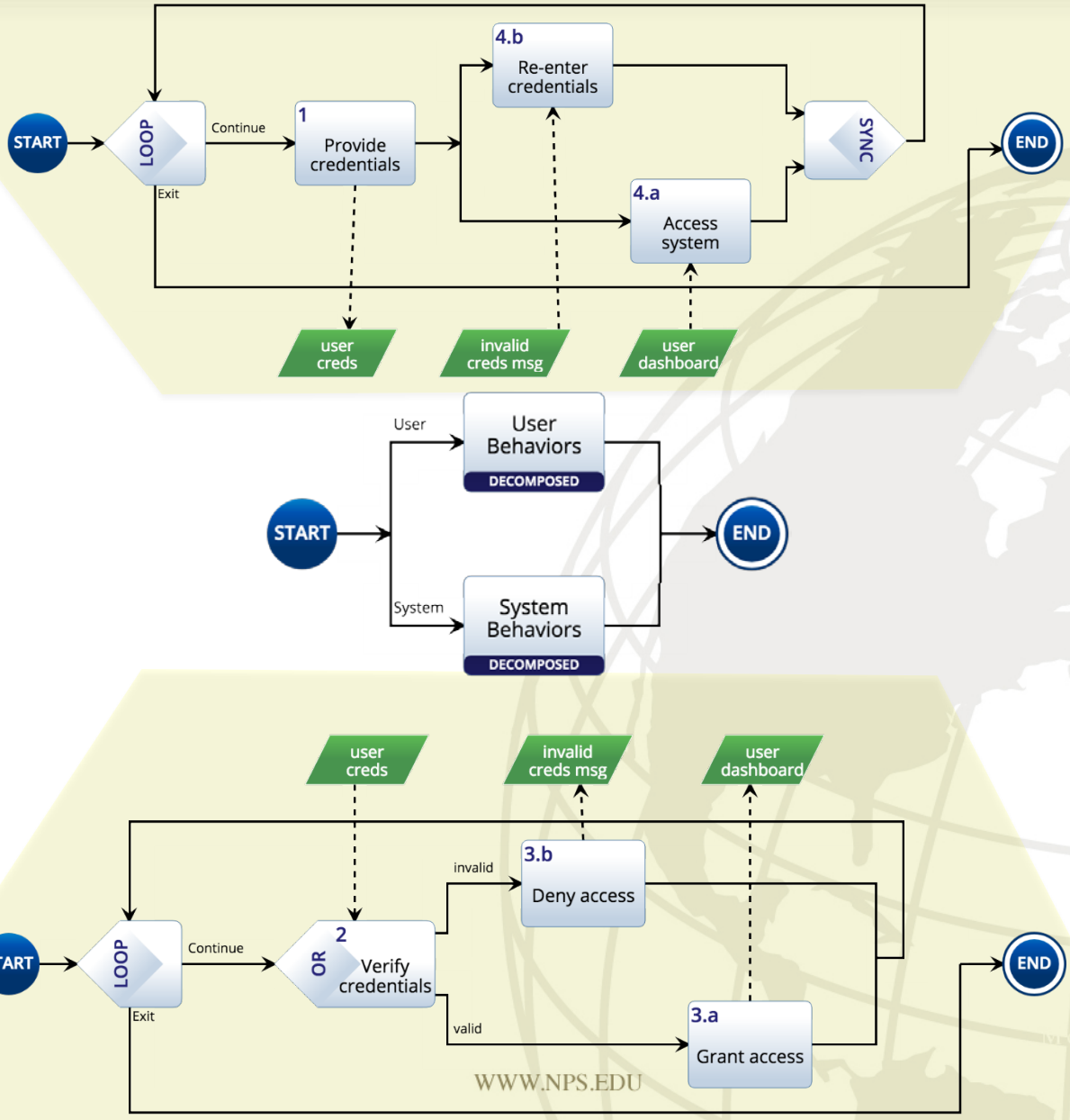
MONTEREY PHOENIX
BEHAVIOR MODELING





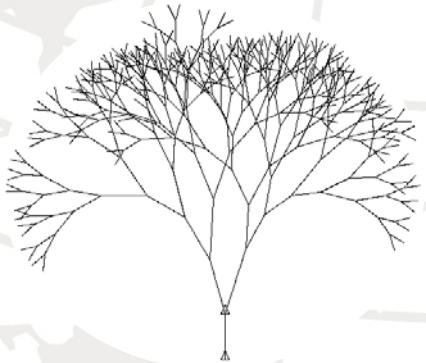
Without interaction constraints: 224 scenarios at scope 3
With interaction constraints: 6 scenarios at scope 3

5. Use abstraction and refinement to manage large models



Way Ahead

- These concepts were distilled from modeling with Monterey Phoenix (firebird.nps.edu)
- Experiment with using these concepts in other behavior modeling languages (e.g., SysML, LML)
- Use MP to expose and prune away negative emergence in behavior models



https://www.cs.unm.edu/~williams/cs257/graphics.html

Questions?

Monterey Phoenix and Related Work:

https://wiki.nps.edu/display/mp

firebird.nps.edu

MONTEREY PHOENIX
BEHAVIOR MODELING

kmgiamma (at) nps.edu